THÈSE

Pour l'obtention du grade de Docteur

Université de Strasbourg

●

Département Mécanique - Équipe MécaFlu
Groupe Instabilités Turbulence Multiphasique
École doctorale MSII

Discipline : Mécanique des Fluides

Raffinement de maillage adapté à la méthode Chimère

Présentée par : Anthony Ponce

Sous la direction de :

Yannick Hoarau, professeur des Universités
Jan Dušek, professeur des Universités

Composition du jury:

Examinateur : M. Yannick Hoarau, Professeur, Univ. de Strasbourg
Examinateur : M. Jan Dušek, Professeur, Univ. de Strasbourg
Rapporteur externe : M. Éric Goncalves, Professeur, ISAE-ENSMA
Rapporteur externe : M. Michel Fournié, Maître de conférence., Univ. de Toulouse

Date de soutenance : 4 juillet 2019

Remerciements

Je tiens tout spécialement à remercier ma mère que je n'ai pas trop ménagé mais qui arrive quand même à me supporter ainsi que mon père qui me soutient plus silencieusement à sa manière... sans eux je ne serais tout simplement pas là...

Je remercie l'énergique Christelle Bernard, catalyseur du début de cette thèse, pour son soutien et sa bonne humeur.

Je remercie également mon tuteur Yannick Hoarau qui m'a accueilli au sein de l'équipe ITD. Merci à lui pour sa patience et son optimisme.

Merci également à la dream team du labo avec en premier chef le seul, l'unique et l'irremplaçable Lord Daniel Durrenberger aka Hanz Haplast ou le nouveau Newton des temps modernes, l'as de la pomme, le tyran des pigeons et bien évidemment le virtuose du travail à la cool ! Merci à lui pour les soirées détente et les sorties sportives ! Merci également à Dorian Pena avec qui il était toujours agréable de développer et de déboguer des scripts utiles ici et à l'autre bout de la planète. Merci aussi à Vincent Léautaud, le pragmatique qui s'ouvre aux possibilités de changement de modèle pour notre système rouillé. Merci aux autres membres de l'équipe, du labo et de passage, Chao-Kun, Ali Alkebsi, Viswa Moturi, Sandra, Pierre, Abder, Madhi pour les bons moments !

Merci à Abdel Azizi pour les pauses discussion. Vivement le miel... en espérant que les abeilles survivent !

Merci à l'équipe de l'association Desclicks - l'Informatique Solidaire à Schiltigheim : Damien Gleitz, Philippe Basley, Adrien Berard, Jean Robot, Alexandre Tsilefski, Alexis Mercier, Jérémy Basley, Régis Hamann, Jean-Marc Heller ; pour m'avoir permis de me sentir un brin utile.

Merci également aux joyeux lurons de l'association RECI, François Berger, Gillou Reinhart et le cartographe en herbe Ben Scherrer avec qui les vendredis soirs ont été bien remplis de jeux qu'ils soient de société ou virtuels et autres discussions sur la vie, l'univers et tout ce qui est.

Merci aux membres de l'AHQCS du groupe compost et jardin, Agnès Fresnoit, Ca-

roline Cavé, Catherine Weckmann, Gaël Jouenne pour les week-end de construction de potager et d'apprentissage commun de la permaculture.

Merci à la FI de Schilick et des alentours, Marc Baader, Guy Desportes, Louisa Krause, Floriane Dupré, Sébastien Mas, Frédéric Staut, Josiane Bigle, Cathou et Jean-Mich, Anthony Bru...

Merci aux super-héroïnes et héros de Résistance à l'Agression Publicitaire Strasbourg Charlotte Ribaute, Baptiste Marlard, Gaétan Liss, Bastien Poix, Danielle Siegler, Julien Guillaume pour l'énergie et la joie partagée pour faire tomber le système publicitaire.

Merci à tous ceux que je n'ai pas cité... et qui, par leur action, par leur présence, leur absence, ont contribué au déroulement et l'achèvement de cette période.

Enfin, je n'allais pas vous oublier ;) un grand merci à ce petit couple détonnant Jérôme et Anna Postler. Vous avez boosté ma barre de vie pendant un mois de folie d'élections. Merci pour les soirées, merci pour les sorties, merci pour les échanges, merci pour l'aide, merci pour le soutien... merci pour tout... merci d'être là... Mais vous le savez, un grand merci pour vous deux c'est juste un doux euphémisme.

Et encore merci à toi Anna.

Table des matières

Introduction

Notre environnement est source d'émerveillement, d'interrogations mais également un puits sans fin d'inspirations. L'observation attentive des phénomènes qui s'y déroulent permet d'extraire des lois qui permettent de comprendre et de prédire le déroulement de ces phénomènes et nous aider à répondre à des questions comme : Comment va évoluer la météo de demain ou le climat dans une centaine d'années ? Comment va s'écouler l'air autour d'une aile d'avion quand celle-ci est givrée ? Les sollicitations éprouvées par les structures de refroidissement au cœur d'un réacteur nucléaire vont-elles mener à sa destruction ? Un des objectifs des sciences est donc de proposer des descriptions notre environnement. Ces descriptions passent par l'écriture d'équations définissant des modèles mathématiques et physiques. Une fois ces modèles définis ceux-ci doivent être confrontés à la réalité afin de pouvoir être validés. Ces validations passent donc par la résolution d'équations sur un domaine de calcul. Malheureusement il est en général impossible de proposer une solution simple. Afin d'obtenir un résultat il est nécessaire de discrétiser le domaine, c'est-à-dire de le découper en petites boites permettant de résoudre nos équations localement. L'ensemble de ces boites appelées mailles sont connectées entre elles afin de paver l'ensemble de l'espace à étudier, ce pavage est appelé maillage. Ces boites sont reliées à leurs voisines et partagent les informations sur leurs frontières permettant ainsi de proche en proche d'obtenir une solution sur l'ensemble du domaine. La précision de la solution obtenue dépend alors des processus physiques œuvrant localement ainsi que de la taille des mailles. Souvent, plus les mailles sont petites, plus les calculs peuvent être précis. Cependant nos calculs étant fastidieux et répétitifs, des ordinateurs les opèrent à la place des humains pour gagner du temps et éviter des erreurs. Les machines vont calculer beaucoup plus rapidement et précisément que nous, cependant elles prennent du temps elles aussi pour calculer. Lors de la définition du maillage il est donc primordial d'opérer un équilibre entre précision et vitesse de calcul.

À la problématique de la définition d'un maillage équilibré initial peut également se greffer celle consistant à modifier celui-ci en cours de calcul afin de s'adapter à une géométrie se déformant. Cette problématique se pose notamment au sein de l'équipe avec l'étude du givrage de profils d'ailes d'avion menée notamment par Dorian Pena [99]. Après avoir calculé l'écoulement aérodynamique autour de la géométrie d'aile, un calcul d'accrétion

de particules de glace est effectué permettant d'obtenir un profil de givre. Ce profil est stocké sous la forme d'une fonction qui, avec la croissance du bloc de givre va avoir pour support un maillage peu adapté pour capter finement l'écoulement proche parois.

Afin de générer un maillage pouvant s'adapter dynamiquement à une géométrie en évolution tout en gardant un équilibre entre densité et qualité de calcul, il est possible d'opter pour un raffinement automatisé du maillage de notre domaine. Ce raffinement permettra ainsi de concentrer de petites mailles dans les zones du domaine nécessitant une grande précision.

Dans ce travail de thèse nous nous intéresserons plus spécifiquement au raffinement de maillage par ajout de nouveaux blocs superposant l'existant (méthode Chimère). Les développements opérés permettront ainsi d'enrichir les fonctionnalités du programme polyvalent de résolution d'écoulements fluides utilisé au sein de l'équipe Instabilités Turbulence Multiphasique (anciennement Instabilités Turbulence Diphasique) du laboratoire ICube de Strasbourg.

Cette thèse s'organise en trois parties principales. La première partie est dédiée à l'étude bibliographique, où nous ferons une revue des différents types de maillages et raffinements possibles. Dans une seconde partie, les outils numériques utilisés sont présentés. Le programme au sein duquel notre raffinement est implanté y est présenté, la méthode des maillages Chimères ainsi que notre méthode de raffinement y sont décrites. La troisième et dernière partie présente plusieurs validations du code sur des écoulements plans d'un fluide newtonien et incompressible dans une cavité entraînée ainsi qu'autour d'un barreau à section carré et d'un barreau à section ronde.

Première partie

Introduction

Chapitre 1

Techniques de Maillages

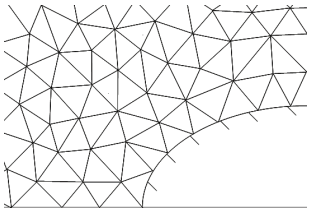
Le monde réel qui nous entoure est décrit par des modèles physiques prenant largement pour hypothèses une continuité temporelle et spatiale des phénomènes. Ces modèles se traduisent par un ensemble d'équations différentielles dont la résolution analytique peut s'avérer extrêmement complexe voire impossible. Pour obtenir une solution permettant de comprendre les processus à l'œuvre il est donc nécessaire de simplifier les modèles par exemple en négligeant certains effets non essentiels. Cependant malgré ces simplifications, les solutions analytiques sont rares et ne répondent pas aux besoins pratiques.

Si une solution analytique à un problème n'est pas disponible, il est possible de mettre à profit la puissance de calcul offerte par les ordinateurs pour l'approcher de manière discrète. Pour ce faire, les équations décrivant le monde réel doivent être discrétisées spatialement et temporellement. La discrétisation temporelle s'appuie le plus souvent sur l'avancement en temps par des méthodes de différences finies. La discrétisation spatiale demande elle une plus grande attention et peut être appréhendée suivant différentes techniques dont certaines seront détaillées par la suite. Le choix des paramètres du schéma temporel d'intégration (implicite, explicite, ordre) sont ensuite fonction des propriétés du maillage.

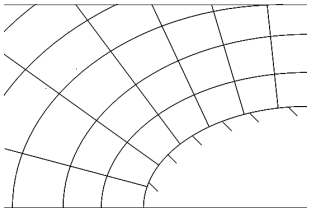
La discrétisation spatiale d'un problème n'est pas l'apanage de la mécanique des fluides. Tous les modèles décrits par des équations aux dérivées partielles peuvent être discrétisés par la mise en place d'un maillage pour obtenir un résultat. C'est l'approche adoptée dans cette thèse.

Depuis la méthode d'Euler (1707-1703) les maillages ont grandement été améliorés et nous détaillons dans la suite les principales familles de techniques. Un maillage a pour objectif de découper l'espace en cellules qui seront le support des calculs. Ces cellules peuvent être identiques sur l'ensemble de l'espace ou allier différents type comme des carrés, triangles, parallélépipèdes et tétraèdres ou des formes polyédriques plus exotiques. Il est également nécessaire de gérer le stockage mémoire de ces cellules pour pouvoir les parcourir et stocker les informations utiles à leur repérage spatial. Pour ceci, deux principales approches peuvent être mise en pratique, l'approche structurée et l'approche non-structurée. La multiplicité des types de maillages est illustrée par quelques exemples

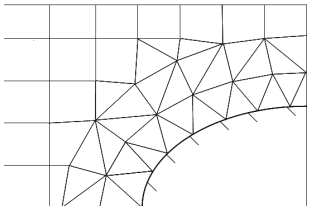
(cf. Mentor Graphics Whitepaper [87]).



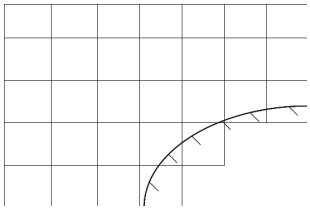
Maillage adapté non structuré



Maillage adapté structuré



Combinaison de maillages



Maillage cartésien non adapté

Figure 1.1 – Exemples de types de maillage traitant la présence d'une frontière

La conception d'un maillage obéit à deux contraintes opposées. La première contrainte est celle de la précision du calcul, en effet plus fine est la résolution du maillage, meilleure sera l'approximation des comportements réels. La seconde contrainte est celle plus terre à terre des ressources matérielles et du temps à disposition. Le temps de calcul est malheureusement une fonction strictement croissante, au mieux linéaire, dépendant du nombre de points du maillage. Un maillage doit donc offrir un bon "rendement" entre précision et temps de calcul. De plus le comportement numérique du maillage est également à prendre en compte lors de sa conception, en effet certains maillages nécessitent plus de temps de calcul du fait d'interpolations, au prix d'une meilleure représentation des frontières et captation des phénomènes physiques. Les prochains points détaillent un aperçu des méthodes actuelles de maillages de l'espace.

1.1 Maillages Cartésiens

Les maillages cartésiens sont en fonctions de la dimension du problème, respectivement composés uniquement de segments, rectangles ou parallélépipèdes rectangulaires. L'espace est tout simplement découpé en une grille d'éléments orthogonaux. Cette approche a l'avantage de ne demander que peu d'effort à l'utilisateur lors du processus de génération du maillage. Les cellules étant par construction toutes alignées et de même type, il est possible de définir un ordre de parcours afin de les numéroter, ce qui fait

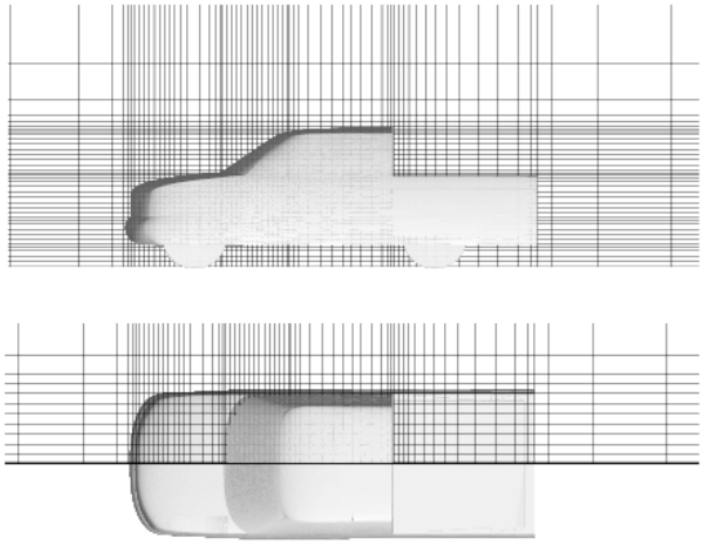


FIGURE 1.2 – Maillage cartésien plus raffiné au niveau des parois

d'un maillage cartésien également un maillage structuré. Un autre avantage d'un maillage cartésien est la faciltié d'implémetation de méthodes telles que différeces ou volumes finis. Enfin il est possible d'adapter la résolution locale du maillage pour obtenir une meilleure précision et ainsi mieux capter la physique du problème. Un exemple de définition de maillage cartésien peut être retrouvé sur la figure du pick-up KALITZIN, IACCARINO et KHALIGHI [68]. Pour conclure sur les avantages des maillages cartésiens, ceux-ci sont faciles à générer, demandent moins d'espace mémoire et sont moins coûteux en temps de calcul que des maillages conformes à la paroi [65, 2].

Le problème des maillages cartésiens est qu'ils interdisent de mailler une surface courbe sans crénelage de celle-ci. Même s'il est possible d'augmenter la résolution pour diminuer "l'effet escalier", supprimer cet effet est impossible. Les simulations autour de géométries courbes à Reynolds élevés, même avec une fine résolution ne peuvent donc pas être menées correctement. Pour contourner ce problème et continuer d'utiliser des maillages cartésiens pour des simulations à Reynolds élevés il est donc nécessaire d'utiliser une méthode de frontières immergées communément appelées *IBM* pour *Immersed Boundary Method*, ou une méthode de *level* set. Ces méthodes permettent de prendre en compte des frontières coupant des cellules. Un dernier problème des maillages cartésiens concerne son raffinement. Il en effet impossible de raffiner seulement localement un maillage cartésien. Soit on divise localement une maille et on répercute cette division suivant les différents axes du problème ce qui peut augmenter de manière sensible le nombre de cellules. Soit on souhaite diviser uniquement localement une maille et donc on doit opter pour un maillage non-structuré. En effet il est impossible d'ajouter localement une maille dans une grille structurée. Les maillages cartésiens ne sont donc pas adaptés pour être raffinés.

1.2 Maillages conformes à la paroi

Les exemples des figures 1.1 et 1.2 montrent que les maillages cartésiens s'adaptent mal à la présence de frontières ce qui peut poser une limitation pour le traitement des simulations à haut Reynolds. En effet pour traiter de tels cas il est nécessaire d'avoir une bonne définition des effets de couche limite. Pour obtenir une telle définition il est donc nécessaire de coller au mieux à la géométrie des corps. Les méthodes de maillages dites conformes à la paroi ou *body fitted* en anglais ont donc été développées dans cette optique. Une bonne partie de ces maillages sont créés à partir de mailles à forme triangulaire. La première étape de la construction du maillage passe donc par la définition surfacique des géométries des corps immergés et des frontières du domaine par la création d'une première rangée de cellules. La seconde étape consiste à mailler le reste du volume du domaine. Pour des géométries planes il est possible d'utiliser une grande variété de type de mailles : hexagones, trapèzes, portion de couronne, polyèdre ou simplement maille cartésienne déformée. En trois dimensions des mailles en forme de tétraèdres sont principalement utilisées. Deux exemples communs de cylindre ou profil d'aile ainsi qu'un troisième de conduit plus complexe sont illustrés respectivement par les figures 1.3 et 1.4. Actuellement les maillages conformes restent majoritaires. Ces maillages offrent de plus la possibilité de définir de manière optimale la position des points à la surface des géométries et se prêtent également bien à la gestion de frontières immergées. Les maillages conformes, en permettant une adaptation fine du maillage des parois et de leur voisinage sont donc parfaitement indiqués pour capter correctement les effets de couche limite. Cette possibilité d'adaptation fine augmente cependant le coût de création et d'automatisation de cette création pour de tels maillages. De plus pour des cas industriels souvent complexes, du fait de la nécessité de contrôler localement la qualité de la résolution du maillage, le développeur doit investir une somme de travail souvent considérable et ceci pour capter fidèlement les effets de couche limites. Un autre souci qui peut être rencontré avec des maillages conformes est la gestion des corps mobiles ou se déformant. Avec de tels mouvements, pour conserver la propriété de conformité à la paroi, il est donc nécessaire de remailler entièrement le domaine ou a minima de procéder à une modification du maillage par déformation de celui-ci et ceci à chaque pas de temps. Ces opérations sont lourdes, impactent sur le temps de calcul et nécessitent une attention particulière de l'utilisateur pour s'assurer que les déformations du maillage n'impactent pas significativement la qualité de la résolution du problème. En effet les déformations de maillages peuvent entraîner des erreurs numériques provenant du cumul d'approximations lors des modifications des distances entre points du maillages et de la nécessité du recours systématique aux interpolations. Les maillages conformes nécessitent également un temps de calcul plus élevé et peuvent être moins robustes comparés à des maillages cartésiens du fait de la nécessité du calcul d'interpolations.

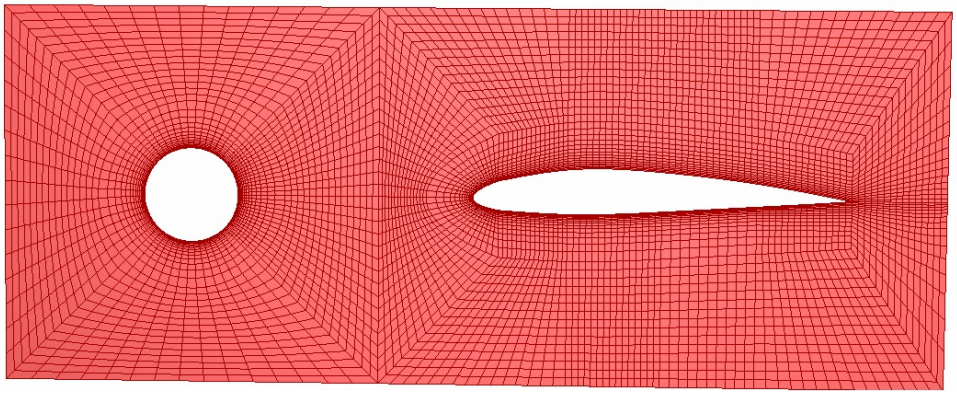


FIGURE 1.3 – Maillages conformes à base de trapèzes

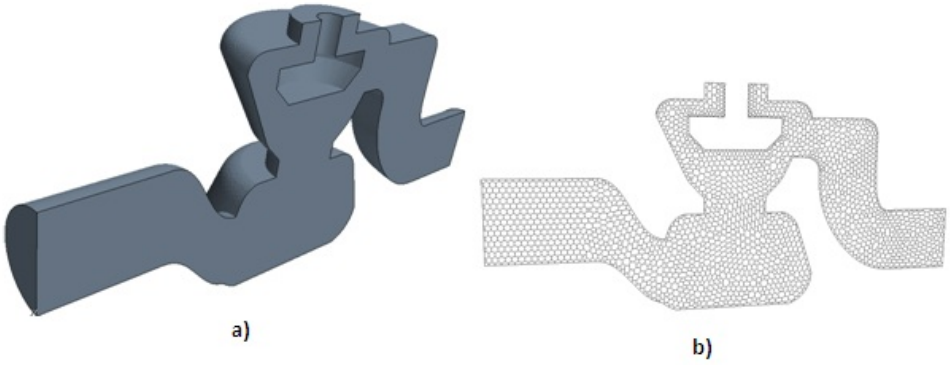


FIGURE 1.4 – Maillage conforme à base de polyèdres

1.3 Maillages structurés

Nous avons déjà évoqué les maillages structurés ou non, dans cette section nous détaillons les maillages dits structurés. Un tel maillage est composé de la répétition d'une maille élémentaire, celle-ci pouvant être carrée, rectangulaire, triangulaire, hexagonale, cubique, parallélépipédique rectangulaire, tétraédrique, hexaédrique ou d'autres types moins répandus permettant de paver l'espace. La répétition d'une cellule identique permet de créer rapidement et facilement un maillage. Il est cependant possible d'adapter la taille de la cellule primaire sans en modifier son type et ceci afin de densifier une zone spécifique du maillage comme on peut le voir sur les figures 1.2.

Ce type de maillage reste tout de même assez rigide et à l'exception de géométries simples, il ne permet qu'exceptionnellement d'être également conforme à la paroi. Les maillages structurés doivent donc régulièrement être utilisés en association avec une méthode *IBM* ou *level set* pour pallier cette limitation.

Sa principale limitation est son adaptabilité coûteuse. En effet si une zone particulière du maillage nécessite une meilleure résolution, l'augmentation locale du nombre de cellules

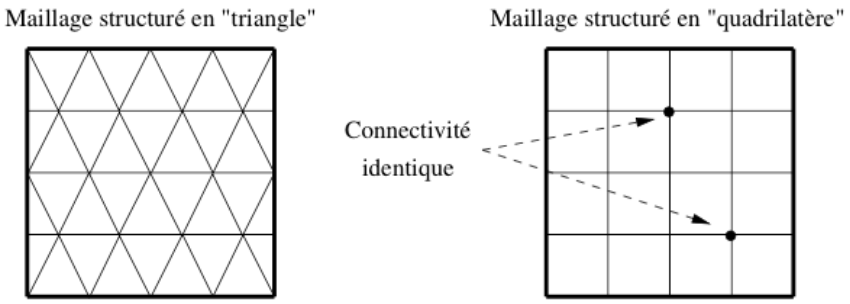


FIGURE 1.5 – Divers maillages structurés

se répercute automatiquement sur plusieurs axes, augmentant mécaniquement le nombre de cellules sur l'ensemble du domaine comme en témoigne le pick-up 1.2. Cette augmentation de cellules est pertinente pour la zone nécessitant une haute densité de mailles, mais l'est grandement moins en général pour l'ensemble des axes affectés. Un maillage structuré est donc peu adapté pour l'étude de cas nécessitant une précision extrêmement localisée.

Enfin, comme dans le cas d'un maillage cartésien, ajouter ou retirer une cellule est impossible. Il est donc exclu d'améliorer localement un maillage ou de supprimer des mailles présentes à l'intérieur d'un corps. Pour ce dernier cas, la charge de l'occultation des cellules inutiles au calcul est à la charge du schéma numérique. De plus l'impossibilité de supprimer des cellules rend les maillages structurés peu propices pour des études complexes où une grande partie du maillage devrait être ignorée comme dans le cas d'une géométrie de poumon fig. 3.4. Sans découpage du domaine en sous-ensembles, un maillage purement structuré perd rapidement en pertinence avec l'augmentation de la complexité de la géométrie [125, 65].

1.4 Maillages non-structurés

Au cours des derniers paragraphes nous avons pu constater que le besoin de raffinement localisé est important et que les maillages cartésiens ne sont pas satisfaisants. Les maillages non-structurés permettent de tirer parti de la puissance croissante des moyens de calculs afin de définir une structure de maillage plus souple mais plus gourmande en stockage et permettant l'augmentation localisée de la résolution des grilles générées. Avec un maillage non-structuré la topologie peut être totalement arbitraire. La cohabitation de multiples types de mailles est possible. Ainsi on pourra trouver des maillages plans non-structurés mélangeant mailles triangulaires et/ou quadrangulaires ou encore en trois dimensions des maillages alliant tétraèdres, prismes, hexaèdres et/ou des pyramides.

La souplesse du maillage non-structuré réside dans le fait que chaque cellule est définie

par ses faces. Chaque face connaît également via la structure de données l'identité des cellules dont elle est le support. Ainsi avec une telle structure de données, il est facile de supprimer une cellule ou d'en ajouter sans complexifier outre-mesure le maillage dans sa globalité. Le raffinement ou dé-raffinement reste ainsi une procédure localisée, contrairement aux maillages structurés et spécifiquement cartésiens. On notera toutefois que pour maintenir une bonne qualité de résolution, il est important de conserver une certaine progressivité dans l'évolution des tailles de mailles au travers du maillage. Une illustration de la capacité de gestion des interfaces avec un maillage non-structuré est illustré par la figure 1.6 issue de Rome [108].

La génération d'un maillage non-structuré est plus technique que celle d'un maillage structuré. Cependant le coût de cette complexité est compensé par la modularité offerte. En effet les maillages non-structurés sont bien indiqués pour générer des maillages conformes à la paroi et sont plus facile à mettre en œuvre pour l'étude de géométries complexes Hassan, Probert et Morgan [56] et Jahangirian et Hashemi [65].

Créer un maillage non-structuré demande un temps de développement qui peut être important. Si ajouter ou supprimer des mailles peut-être réalisé sans grand effort, raffiner une maille nécessite une attention particulière pour gérer la modification des connectivités entre les cellules et les nouveaux cas topologiques apparaissant aux nouvelles interfaces ainsi créées. Le schéma numérique devra également être adapté pour prendre en compte correctement ces modifications de configurations.

Enfin un maillage non-structuré requiert une quantité de mémoire nettement supérieure du fait de la nécessité de stockage de la structure de données pour chaque cellule. Ce besoin peut devenir un frein à la simulation de géométrie complexes sur des machines de relativement "faible" puissance. Ce frein peut être contourné par diverses méthodes comme l'utilisation de niveaux de maillages plus grossiers ou éviter le stockage de grandeurs et les retrouver "à la volée" lors de l'exécution du code. Pour conclure, on notera qu'une fois que le coût de mise en place des outils de raffinement est payé, l'automatisation de l'adaptation du maillage est relativement peu onéreuse.

1.5 Maillages conformes

La propriété de conformité d'un maillage assure que chaque nœud d'un maillage soit disposé à une extrémité de maille. Les maillages cartésiens et structuré sont donc conformes par construction. Les maillages non-structurés peuvent être conformes ou non. La non-conformité d'un maillage nécessite la prise en compte de traitements spécifiques

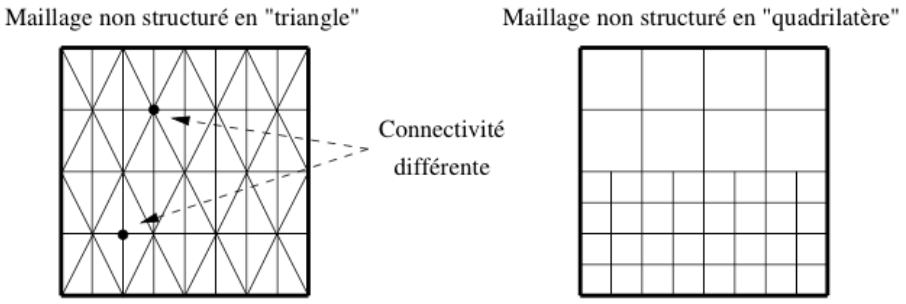


FIGURE 1.6 – Divers maillages non structurés. À gauche maillage conforme, à droite maillage non conforme

aux parois.

1.6 Conclusion

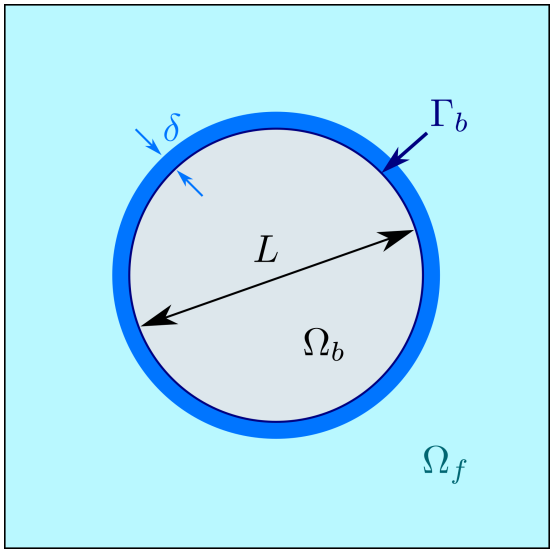
Le maillage "parfait" n'existant pas, l'utilisateur pourra utiliser les différentes techniques de maillages de base à sa disposition pour définir un maillage adapté au problème qu'il souhaite traiter. Le temps de création d'un maillage est cependant un point de tension important pour une étude et il est plus intéressant d'investir dans un processus de raffinement de maillage automatique. En effet avec un raffinement automatique, l'utilisateur s'affranchit du processus itératif de création d'un maillage permettant de traiter des écoulements complexes.

Chapitre 2

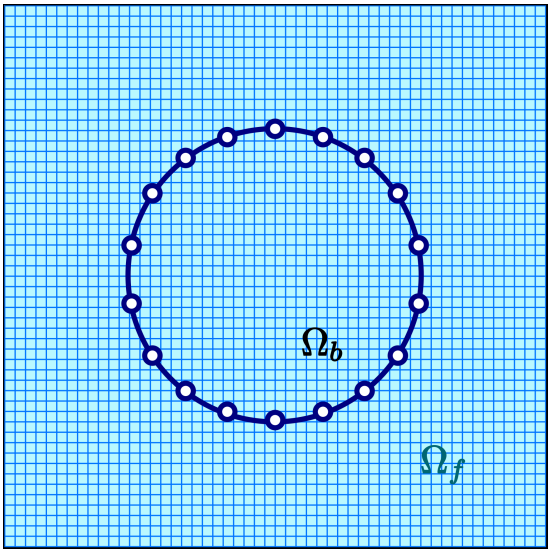
Méthode des frontières immergées

La méthode des frontières immergées, connue sous l'acronyme IBM pour Immersed Boundary Method, offre la possibilité d'étudier des géométries complexes sur des maillages notamment cartésiens. Pour gérer la frontière la méthode introduit un terme de forçage dans la discrétisation, aux frontières de l'élément immergé ou directement dans les équations. Ce forçage permet de s'affranchir de la limitation de frontière crénelée et non idéale à la description des couches limites inhérente à l'utilisation d'une géométrie ayant pour support un maillage cartésien. Les conditions limites peuvent donc ainsi ne plus être supportées par le maillage. De plus, moyennant une grille adaptée, cette méthode permet de gérer les parois mobiles sans nécessiter un maillage mobile de type chimère.

Différentes variantes de la méthodes IBM sont décrites en détails dans Mittal et Iaccarino [89] ou Merlin [88], que nous résumons partiellement dans cette partie.



(a) Schéma d'un cylindre de volume Ω_b, longueur caractéristique L, frontière Γ_b enveloppé d'une couche limite d'épaisseur δ immergé dans un fluide occupant Ω_f



(b) Schéma du même cylindre immergé dans un maillage cartésien sur lequel les équations régissant le fluide sont discrétisées

Figure 2.1 – Un cylindre immergé

Considérons l'étude de l'écoulement autour d'un cylindre comme sur la figure 2.1(a).

Classiquement pour capter correctement la physique proche paroi, un maillage conforme à la paroi structuré ou non est défini. Une fois le maillage surfacique de la paroi défini, celui du domaine peut-être réalisé. Le maillage surfacique sert de support à la définition des conditions limites. Avec un schéma aux différences finies, sur un maillage structuré, la forme différentielle des équations peut être modifiée pour être écrite en système de coordonnées curvilignes suivant les lignes du maillage [43]. Le maillage étant conforme à la paroi, la discrétisation des équations transformées est aisée. Avec un schéma volumes finis c'est l'équation sous sa forme intégrale qui est discrétisée. La géométrie du maillage est alors directement intégrée dans la discrétisation.

Considérons maintenant la même étude qu'évoquée au paragraphe précédent mais s'appuyant maintenant sur un maillage cartésien. Le maillage étant maintenant non conforme, la définition des parois de notre cylindre entrecoupe les cellules de la grille. Afin de prendre en compte la présence de la paroi, il est donc nécessaire d'adapter les équations du système sur les cellules voisines de la paroi pour y intégrer les conditions limites. Ces modifications seront détaillées par la suite. Avec une telle procédure, aucune transformation spatiale ou d'opérateurs complexes de discrétisation des équations n'est nécessaire que l'étude soit menée en différences finies ou en volumes finis.

Comparons maintenant les avantages et inconvénients des deux approches. Premièrement commençons par le support du calcul, le maillage. Le but d'un maillage est d'offrir un bon rapport entre une bonne résolution nécessaire à la captation fidèle des phénomènes physiques et une taille réduite afin de répondre aux exigences de temps de calcul et d'espace de stockage numérique nécessaire lors de la simulation. Les deux contraintes précédemment évoquées sont en compétition. À l'exception des cas très simples, réduire le nombre de points ou cellules d'un maillages affecte la précision ou les propriétés de convergence du solveur [43]. Créer des maillages conformes à la paroi efficaces est donc un processus technique et itératif coûteux en temps d'élaboration et dont la complexité est fonction de celle de la géométrie. Pour alléger la complexité de cette création il est commun, pour des maillages structurés, d'utiliser des techniques de division du maillage [104] comme la méthode chimère. Ces techniques reportent cependant la complexité de la création du maillage sur la résolution du problème en nécessitant notamment une bonne gestion de la communication entre les interfaces des différents blocs. La génération de maillages non-structurés quant à elle, même si elle se prête plus facilement à la gestion de géométries complexes, voit en général la qualité des maillages générés diminuer avec la complexité de la géométrie. À l'inverse, avec une méthode IBM, la génération d'un maillage est facilitée par la possibilité d'utiliser un maillage cartésien. Un tel maillage est par construction plus aisé, sa complexité et sa qualité n'étant pas ou peu affectée par la complexité de la géométrie à étudier.

Le second avantage de la méthode IBM non négligeable est son utilisation pour gérer des géométries en mouvement. En effet avec des maillages conformes, structurés ou non, gérer le mouvement va se traduire par la nécessité de remailler soit en déformant le maillage initial, soit en déplaçant le maillage ainsi qu'en projetant la solution sur le nouveau maillage [124] ou en incorporant la vitesse du maillage dans les termes advectifs. Ces manipulations nécessaires peuvent nuire à la simplicité, précision ou robustesse du schéma de résolution et impactent également le temps de calcul spécialement si de nombreux éléments de maillage doivent être déplacés. Finalement, malgré les progrès constants en matière de gestion de la mobilité sur maillages structurés [14, 107, 124], la méthode IBM, par sa simplicité, reste un choix judicieux.

Pour tempérer les avantages de la méthode IBM, notons que l'imposition des conditions limites requiert une attention particulière et peut influer sur la précision ainsi que sur la conservation des propriétés du schéma numérique. L'utilisation de maillages conformes permet au contraire un meilleur contrôle de la densité des cellules au voisinage des parois facilitant le traitement de simulations à haut Reynolds. Avec un maillage cartésien, améliorer la résolution dans une zone implique une augmentation globale du nombre de mailles réparties sur l'ensemble de celui-ci. Cette augmentation n'augmente cependant pas la complexité même du maillage et le coût d'un calcul en maillage cartésien reste plus faible du fait de l'absence de transformations de coordonnées des équations.

2.1 Imposition des conditions limites

Comme indiqué précédemment, l'imposition des conditions limites requiert une attention particulière au bon développement d'une méthode IBM. Sa mise en œuvre distingue les différentes méthodes. Soit un écoulement incompressible autour du cylindre fig. 2.1(a), régi par les équations de Navier-Stokes (NS) incompressibles :

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} + \frac{1}{\rho}\nabla p - \frac{\mu}{\rho}\nabla^2 \boldsymbol{u} = 0 \qquad \text{et} \tag{2.1}$$

$$\nabla \cdot \boldsymbol{u} = 0 \qquad \text{dans } \Omega_f$$

avec

$$\boldsymbol{u} = \boldsymbol{u}_\Gamma \qquad \text{sur } \Gamma_b, \tag{2.2}$$

où $\boldsymbol{u}$ est la vitesse, p la pression, et ρ et μ respectivement la masse volumique et la viscosité

du fluide. Le solide occupe un volume Ω_b, de frontière Γ_b. Le domaine fluide est noté Ω_f. Pour simplifier, les limites du domaine du calcul sont omises. Pour simplifier l'écriture, on rassemble les équations de mouvements et de continuité en :

$$\mathcal{L}(\underline{U}) = 0 \quad \text{dans} \quad \Omega_f \tag{2.3}$$

$$\underline{U} = \underline{U}_\Gamma \quad \text{sur} \quad \Gamma_b \tag{2.4}$$

avec $\underline{U} = (\boldsymbol{u}, p)$ et $\mathcal{L}$ l'opérateur représentant le système d'équations de *NS* présenté dans l'équation 2.1. Rappelons qu'avec un tel système *NS* incompressible, la pression est déterminée par la contrainte de continuité. L'équation de continuité est donc considérée pour la pression, comme une équation implicite.

Classiquement l'équation (2.3) est discrétisée sur un maillage conforme aux parois, sur lequelles (2.4) est imposée. Avec une méthode *IBM*, l'équation 2.1 est discrétisée sur un maillage cartésien non conforme. Les conditions limites sont alors prises en compte par une modification de (2.3). Généralement, cette modification est opérée via l'introduction d'un terme source afin de prendre en compte la présence d'une frontière. Ce terme source ou fonction de forçage peut-être implémenté par deux méthodes différentes discutées par la suite. Ces deux techniques permettent de définir deux classes de méthodes *IBM*.

La première technique consiste à insérer au sein de l'équation de continuité (2.3), une fonction de forçage $\underline{f}_b$ menant à l'équation (2.5) suivante sur l'ensemble du domaine :

$$\mathcal{L}(\underline{U}) = \underline{f}_b, \text{ sur } (\Omega = \Omega_f + \Omega_b), \quad \underline{f}_b = (\boldsymbol{f}_m, f_p) \tag{2.5}$$

où $\boldsymbol{f}_m$ et f_p sont respectivement les fonctions de forçage appliquées aux moments et à la pression. L'équation (2.5) est ensuite discrétisée sur un maillage cartésien comme suit et résolue sur l'ensemble du domaine Ω :

$$[L]\{\underline{U}\} = \{\underline{f}_b\}. \tag{2.6}$$

La seconde technique consiste à discrétiser (2.3) en omettant d'abord la présence de frontières immergées. On obtient donc le système $[L]\{\underline{U}\} = 0$ sur Ω. La discrétisation au sein des cellules voisines de la frontière immergée est dans un second temps ajustée pour en prendre compte menant à l'écriture du système modifié $[L']\{\underline{U}\} = \{\underline{r}\}$. Dans cette dernière équation $[L']$ est l'opérateur discret modifié et $\{\underline{r}\}$ représente l'ensemble des termes représentant les conditions limites de la frontière immergée. Le système précédent

peut être reformulé de la manière suivante :

$$[L]\{\underline{U}\} = \{\underline{f'}_b\}, \tag{2.7}$$

$$\text{où } \{\underline{f'}_b\} = \{\underline{r}\} + [L]\{\underline{U}\} - [L']\{\underline{U}\} \tag{2.8}$$

La première technique appelée *forçage continu* intègre la fonction de forçage avant la discrétisation alors que la seconde technique appelée *forçage discret* l'introduit après. Le forçage continu est attractif car permet de s'affranchir de la discrétisation spatiale sous-jacente au problème alors que le forçage continu dépend très fortement de celle-ci. Cependant cette forte dépendance offre également l'avantage de pouvoir contrôler plus finement la précision, la stabilité ainsi que les propriétés de conservation du solveur numérique. Ces deux méthodes sont détaillées plus avant dans les sections suivantes.

2.2 Forçage continu

Les parois élastiques et rigides nécessitent des traitements spécifiques lorsqu'on emploie une méthode IBM avec forçage continu. Ces deux types de parois sont traités séparément par la suite.

2.2.1 Parois élastiques

Initialement développée par Peskin [1, 101] pour une simulation d'écoulement sanguin dans un contexte de contraction musculaire du cœur, la première méthode IBM est donc bien indiquée pour la prise en compte de parois élastiques. L'écoulement du fluide est ici régi par les équations NS incompressibles résolues sur une grille cartésienne stationnaire. La frontière immergée est modélisée par un réseau de fibres élastiques. La position des fibres est localisée de manière lagrangienne par un ensemble de points sans masse dont la vitesse est asservie à la vitesse locale du fluide u. Les coordonnées $\boldsymbol{X}_k$ du $k^{\text{ème}}$ point lagrangien sont déterminées par l'équation

$$\frac{\partial \boldsymbol{X}_k}{\partial t} = \boldsymbol{u}(\boldsymbol{X}_k,\, t). \tag{2.9}$$

La loi de Hooke relie la déformation de la paroi à la force $\boldsymbol{F}$. L'effet de la frontière immergée sur le fluide environnant est principalement déterminé par un terme de forçage présent dans l'équation de la quantité de mouvement formulé de la manière suivante :

$$\boldsymbol{f}_m(\boldsymbol{x},\, t) = \sum_k \boldsymbol{F}_k(t)\delta(|\boldsymbol{x} - \boldsymbol{X}_k|), \tag{2.10}$$

où δ est la fonction de Dirac approximée par une fonction lisse de support s'étendant sur plusieurs mailles. Les fibres étant dans le cas général non alignées sur le maillage carté-

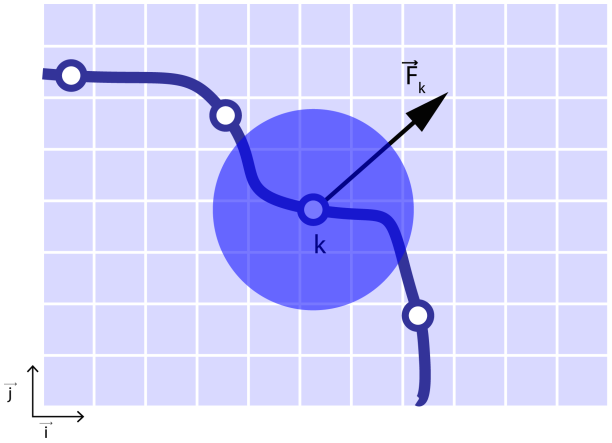


FIGURE 2.2 – Zone de forçage. Image reproduite d'après MITTAL et IACCARINO [89]

Transfert du terme de forçage $\boldsymbol{F}_k$ depuis le point lagrangien vers les points du fluide alentours. La zone ombrée montre l'extension de la distribution de la force.

sien, le terme de forçage est distribué sur les cellules au voisinage de chacun des points lagrangiens comme indiqué sur 2.2 en l'imposant dans l'équation de la quantité de mouvement aux nœuds du maillage se trouvant dans le support de la fonction d approximant la fonction de Dirac. Le terme de forçage est donc réécrit pour chaque point $\boldsymbol{x}_{ij}$:

$$\boldsymbol{f}_m(\boldsymbol{x}_{i,j},\ t) \ = \ \sum_k \boldsymbol{F}_k(t) d(|\boldsymbol{x}_{i,j} - \boldsymbol{X}_k|). \tag{2.11}$$

De manière similaire la vitesse du fluide dans l'équation (2.9) est lissée à l'aide de la même fonction de distribution d. Le choix de cette fonction est un élément déterminant de la méthode. La littérature est riche de multiples fonctions [21, 75, 1, 109]. BEYER [21] ont montré la faisabilité de la construction d'une fonction préservant la discrétisation spatiale pour un problème unidimensionnel simple. Les méthodes relevant de cette catégorie ont largement été utilisées pour des problèmes variés comme la mécanique cardiaque [101], la dynamique cochléaire [21], la locomotion aquatique animale [42], la dynamique des bulles [130] ou encore l'écoulement à travers des filaments flexibles [142].

2.2.2 Parois rigides

La méthode précédente adaptée aux parois élastiques pose en général des problèmes de définitions pour l'application à des écoulements autour de parois rigides. Ces problèmes peuvent être dépassés par deux approches. La première consiste simplement dans la formulation précédente à considérer les fibres élastiques comme extrêmement rigides. La seconde approche consiste à considérer le corps rigide comme fixé à une position d'équilibre [20, 75] par un ressort exerçant une force de rappel $\boldsymbol{F}$:

$$\boldsymbol{F}_k(t) \ = \ -\kappa(\boldsymbol{X}_k \ - \ \boldsymbol{X}_k^e(t)) \tag{2.12}$$

où κ est un coefficient de raideur positif et $\boldsymbol{X}_k^e$ est la position d'équilibre du $k^{\text{ème}}$ point lagrangien. L'imposition fine de la condition limite sur la surface de la frontière immergée nécessite l'imposition de valeurs de κ élevées. Cette imposition conduit à un système extrêmement rigide d'équations soumises à d'importantes contraintes de stabilité [75, 117]. De plus, l'utilisation de faible valeur de κ peut conduire à une déviation importante de la position d'équilibre, comme dans des simulations à faibles Reynolds de sillages à l'arrière d'un cube [75].

Cette dernière approche peut-être considérée comme une application particulière du modèle développé pour la simulation d'écoulements autour de corps rigides construite par Goldstein, Handler et Sirovitch [53]. Dans ce modèle en effet, le terme de forçage est écrit comme suit :

$$\boldsymbol{F}(t) \;=\; \alpha \int_0^t \boldsymbol{u}(\tau)d\tau \;+\; \beta \boldsymbol{u}(t), \tag{2.13}$$

avec les paramètres α et β déterminés empiriquement pour une description la plus adéquate des conditions limites à la frontière immergée. L'objectif initial de (2.13) était de permettre un contrôle rétroactif de la vitesse au voisinage de la surface [53]. Cette forme d'écriture peut également être interprétée plus physiquement comme un oscillateur amorti [62]. Cette méthode a été utilisée avec succès pour des écoulements turbulents à faible Reynolds au sein d'un canal rainuré ainsi que pour l'amorce de l'écoulement à Reynolds modéré autour d'un cylindre [53]. Les simulations pour des nombres de Reynolds réduits sont en général concluantes, cependant comme précédemment, l'imposition précise des conditions limites pour des Reynolds élevés requiert de grandes valeurs de α et β pouvant mener à des problèmes de stabilité.

Une dernière méthode, différente par son approche consiste à considérer le solide comme un milieu poreux régit par les équations de NS/Brinkman Brinkmann [23]. Cette méthode développée par Angot, Bruneau et Fabbrie [4] et Khadra, Angot et Parneix [70] a notamment été utilisée pour le calcul d'écoulements autour d'un cylindre pour des Reynolds atteignant 200 ainsi que d'une marche descendante [70].

2.2.3 Conclusion

Le forçage continu est une méthode élégante pour gérer les écoulements autour de frontière immergées, notamment élastiques. C'est une méthode relativement facile à implémenter restant proche de la physique qui en fait un choix apprécié en biologie [21, 42, 101] et en écoulement multiphasique [130] où les parois élastiques sont monnaie courante. Cependant, le comportement des termes de forçage lorsque les corps sont considérés comme rigides pose de nombreux problèmes. Ces problèmes peuvent partiellement être contournés par des modèles simplifiés imitant le comportement des parois rigides sur l'écoulement. Cependant ces simplifications se font au détriment de la précision et de la

stabilité. De plus la fonction de distribution d nuit également à la définition précise de la localisation des parois. Ces contraintes font que les écoulements à Reynolds élevés sont difficiles à réaliser avec un tel forçage. Enfin, le forçage continu possède un dernier talon d'Achille qui tient en la nécessité de résoudre l'écoulement à l'intérieur du solide immergé. Lorsque le maillage nécessite d'être plus raffiné à cause d'une géométrie complexe ou d'un nombre de Reynolds plus élevé, la densification du maillage en résultant augmente sensiblement le nombre de calculs et le temps de résolution de l'écoulement, freinant de fait les études à mener.

2.3 Forçage discret

2.3.1 Imposition indirecte des conditions limites

Certains problèmes unidimensionnels permettent de dériver formellement la fonction de forçage imposant une condition limite à la paroi [21], mais ce n'est pas dans le cas général. Plus spécifiquement avec des équations NS, il est impossible de le faire analytiquement sur tout le domaine. Il en résulte que l'ensemble des méthodes abordées au paragraphe précédent nécessitent des modèles simplifiés pour l'implantation des fonctions de forçage. Pour éviter ces simplifications, Mohd-Yusof [93] et Verzicco et al. [133] ont construit une méthode permettant d'extraire le terme de forçage directement de la solution numérique, pour laquelle une valeur prédictive peut-être déterminée. En reprenant la discrétisation des équations NS, sans aucune altération pour prendre en compte les parois immergées et avec des notations identiques à la section 2.2, le système $[L]\{\underline{U}^*\} = 0$, où $\{\underline{U}^*\}$ représente alors une prédiction du champ de vitesse qui est résolu à chaque pas de temps. La fonction de forçage $\{\underline{f}'_b\}$ présente dans l'équation (2.8) est alors définie par

$$\{\underline{f}'_b\} \approx \{\underline{r}\} + [L]\{\underline{U}^*\} = \{\underline{r}\} - [L']\{\underline{U}^*\} \tag{2.14}$$

où $\{\underline{r}\} = \{\underline{U}_\Gamma - \underline{U}^*\}d(|\boldsymbol{X}_k - \boldsymbol{x}_{i,j}|)$ et $[L'] = [L] + ([I] - [L])\delta(|\boldsymbol{X}_k - \boldsymbol{x}_{i,j}|)$, $[I]$ représentant la matrice identité. Comme précédemment, la fonction delta de Dirac est remplacée par une distribution lissée d et l'équation (2.8) peut être réécrite comme suit

$$[L]\{\underline{U}\} = \{\underline{U}_\Gamma - \underline{U}^*\}d(|\boldsymbol{X}_k - \boldsymbol{x}_{i,j}|) + [L]\{\underline{U}j\}d(|\boldsymbol{X}_k - \boldsymbol{x}_{i,j}|). \tag{2.15}$$

La formulation précédente correspond formellement à l'imposition de la condition limite au point lagrangien $\boldsymbol{X}_k$ de la surface immergée. Le principal avantage de l'approche discrète est l'absence de paramètres empiriques provenant de l'utilisateur permettant ainsi de s'affranchir des contraintes de stabilités numériques en résultant. En contre-parti, l'utilisation d'une fonction de lissage contribue au floutage de la paroi. Cette méthode a été utilisée pour la résolution de nombreux problèmes comme un écoulement turbulent au

sein d'un moteur à combustion [133], des écoulements plans [10] ou tridimensionnels [132] autour de corps escarpés ou également dans un réservoir cylindrique agité [63].

2.3.2 Imposition directe des conditions limites

Nous avons vu précédemment que les méthodes IBM sont adaptées à des contextes de simulations à faible Reynolds. Cependant elles le sont moins pour des Reynolds plus élevés car l'utilisation de fonctions de lissage rend les parois diffuses et donc inappropriées à une définition nette requise pour de telles simulations. D'autres approches ont donc été introduites pour permettre une définition précise des frontières en s'affranchissant des termes dispersifs. Ces approches s'opèrent généralement par une modification de la prise en compte des frontières immergées permettant l'imposition de conditions limites de manière plus explicite. Deux approches sont évoquées ci-après.

Cellule fantôme en différences finies ou volumes finis Le principe de cette méthode abrégée en *GCM* pour *Ghost Cell Method* est d'échanger les fonctions de forçages précédemment évoquées par une interpolation [83, 41] dans les équations NS. Du fait de sa construction par interpolation on retrouve également cette méthode sous le nom de méthode de reconstruction (*reconstruction method*) ou d'interpolation [50, 51, 35, 65]. Avec cette méthode le champ de vitesse proche de la paroi est reconstruit via une formule d'interpolation en remplacement des équations de transport. Contrairement aux méthodes précédentes notamment celles utilisant des forçages continue où il était nécessaire de calculer un écoulement au sein de la zone solide, ici l'écoulement est seulement calculé sur la partie du domaine correspondant à la zone fluide. Ceci permet donc d'économiser de nombreux calculs, de gagner en temps d'exécution et de s'affranchir de descriptions du problème détachés de la physique de celui-ci. Dans cette méthode, les cellules fantômes sont celles présentes à la limite intérieure de la paroi solide et possédant une cellule adjacente dont une partie superpose le domaine fluide. La figure 2.3 illustre une cellule fantôme G. Le processus clé de la méthode réside maintenant dans la définition de l'interpolation de chaque valeur générique de champ ϕ_g au point fantôme G. Pour définir cette interpolation, plusieurs méthodes existent en fonction du maillage en présence, des schémas utilisés et de l'ordre d'intégration souhaité. Ces interpolations peuvent donc être linéaires (Ferzinger et Perić [43]), bilinéaires, quadratiques [63, 49, 83] ou mélangeant linéaire dans une direction et quadratique dans une autre (Ferzinger et Perić [43]). Dans tous les cas l'expression de la valeur de ϕ_g localisée au point G peut s'écrire sous la forme

$$\phi_G = \sum \omega_i \phi_i, \tag{2.16}$$

où la somme prend en compte l'ensemble des points nécessaires à l'interpolation, autres points compris et ou les poids ω_i sont fonction de la géométrie. L'équation précédente est donc l'équivalent de (2.8) pour les cellules fantômes et peut être résolue simultanément avec l'équation de NS discrétisée sur le domaine fluide. Cette méthode à été mise en pratique avec succès pour résoudre divers écoulements comme dont un écoulement compressible autour d'un cylindre et d'une aile [49] à des nombres de Reynolds allant jusqu'à $O(10^5)$, de la propulsion aquatique [89], un écoulement dans un passage serpenté [62] ou encore un écoulement turbulent autour d'un pick-up [68].

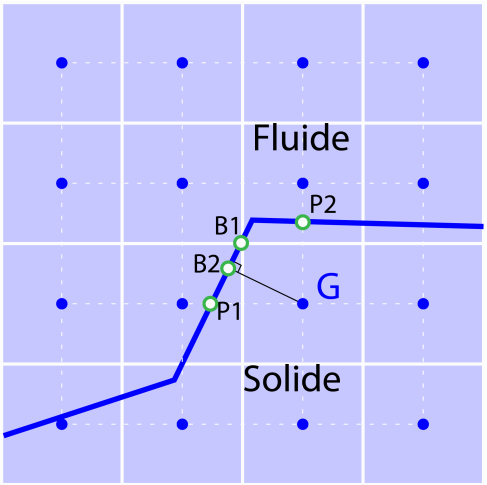


FIGURE 2.3 – Représentation des points aux environs d'une surface immergée pour l'approche par cellules fantômes. G est le point fantôme et B_i et P_i sont les points où la condition limite peut être imposée. B_1 est sur la paroi, équidistant de P_1 et P_2 et B_2 est le projeté orthogonal de G sur la paroi. Image reproduite d'après MITTAL et IACCARINO [89]

.

Cellule coupée en volumes finis L'ensemble des méthodes IBM présentées précédemment ne préservent pas les propriétés de conservativité du schéma numérique sur les cellules voisines des parois. La stricte conservation de masse et de moment peut être obtenu localement et globalement par l'usage de méthode volumes finis. C'est à cet effet que l'approche par cellule coupée a été introduite [30, 65]. Avec cette méthode développée sur des maillages cartésiens, chaque cellule chevauchant une frontière est localisée et les intersections de ses faces avec celle-ci sont calculées. Ces intersections permettent de retirer la partie de la cellule résidant sur le domaine solide. Les cellules dont le centre réside dans le domaine solide sont fusionnées avec les cellules voisines. La définition de la paroi est donc approchée par des trapèzes [140] comme indiqué sur la figure 2.4(a). D'autres configurations plus fines mais plus complexes peuvent également être mises en place.

Une estimation des flux diffusifs, convectifs ainsi que du gradient de pression sur les faces des cellules est nécessaire à la discrétisation des équations de NS en volume finis.

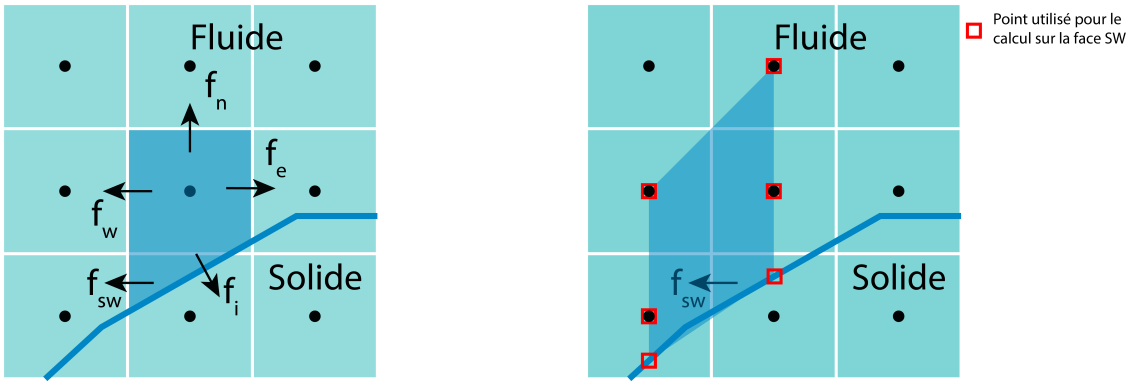


FIGURE 2.4 – **(a)** Volume fini trapézoïdal formé près de la frontière immergée pour laquelle f représente le flux à travers une face. **(b)** Région d'interpolation et points utilisés pour approximer le flux f_{sw} sur la face sud-ouest du trapèze. Image reproduite d'après MITTAL et IACCARINO [89]

La difficulté réside dans l'évaluation de ces grandeurs sur des cellules trapézoïdales. YE et al. [140] ont proposé d'écrire une variable ϕ sous la forme d'un polynôme du second degré dans une zone déterminée puis d'évaluer les flux f en utilisant cette même fonction. Concrètement, sur la figure 2.4(b) l'approximation du flux f_{sw} se base sur une fonction ϕ dépendant linéairement des abscisses et quadradiquement des ordonnées :

$$\phi = C_1 xy^2 + C_2 y^2 + C_3 xy + C_4 x + C_5 y + C_6, \tag{2.17}$$

avec les six coefficients inconnus C_i exprimés en fonction des valeurs de ϕ aux six points mis en avant en rouge sur la figure 2.4(b). Le flux f_{sw} est exprimé de manière similaire à l'équation 2.16. Les flux f_e et f_i sont obtenus de manière analogue. Ce schéma de discrétisation précis au second ordre permet de conserver exactement la masse ainsi que la quantité de mouvement sans dépendance à la résolution du maillage.

Cette dernière méthode a principalement été mise en pratique sur des cas plan pour de multiples simulations telles que des écoulements avec des frontières fixes ou mobiles [92], des feuilles battantes [91], des objets en chute libre dans un fluide [90] et des jets dirigés par une membrane [131]. Le passage de cette méthode en 3 dimensions est cependant complexe du fait de la multitude de configurations de découpage en résultant ainsi que de la difficulté découlant de la discrétisation des équations NS sur des polyèdres. Cette méthode est donc à la limite de la transformation du maillage cartésien en un maillage conforme, comme constaté par BERGER et AFTOSMIS [18].

2.4 Conclusion

Les méthodes détaillées dans la présente section introduisent les conditions limites de manière directe au sein des équations discrétisées. Le forçage discret dépend donc profondément de la méthode de discrétisation et reste moins évident que le forçage continu. Cependant le coût de leurs mises en œuvre permet d'approcher de manière plus exacte la définition des parois ce qui en fait des méthodes plus adaptées aux études à Reynolds élevés. De plus ces méthodes n'introduisent pas de contraintes additionnelles de stabilités dans la représentation des frontières. Enfin ces méthodes permettent de s'affranchir de calculs non physiques à l'intérieur des solides en séparant les équations des nœuds fluides de celles des nœuds solides. Cet avantage est d'autant plus important pour les calculs à hauts Reynolds nécessitant des maillages finement quadrillés. À contrario, l'application de ces méthodes à des frontières mobiles est moins aisée car il est alors généralement nécessaire d'ajouter une condition limite de pression sur la paroi immergée comme dans Udaykumar et al. [129], condition inutile pour des méthodes à forçage continu.

Chapitre 3

Raffinement de Maillages

Les simulations numériques sont des processus coûteux en temps de création et d'exécution. Un équilibre doit être arbitré sur la densité du maillage qui doit obéir à deux contraintes opposées, fournir un support de calcul le plus précis possible et permettre une résolution la plus rapide possible. Le raffinement de maillage automatique est une solution permettant de répondre à ces deux impératifs. Les premières techniques de raffinement de maillages ont été introduites dans les années 1980 avec les travaux de Berger et Colella [17], Beyer et Leveque [19], [103] pour des maillages cartésiens.

Dans cette section, nous ferons une revue des principales familles de techniques de raffinement de maillage.

3.1 Méthodes Adaptatives

Le principe des méthodes adaptatives est de modifier le maillage original afin de l'adapter localement. Nous présentons ici plusieurs stratégies d'adaptation locale de maillage comme illustré dans la figure **??**.

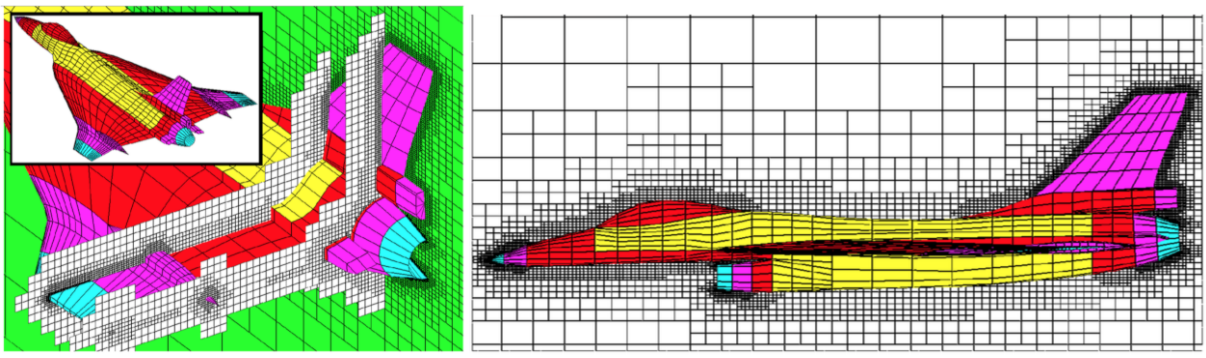


Figure 3.1 – Maillage cartésien raffiné autour d'un F16XL issu de [126]

3.1.1 Raffinement local de mailles (h-raffinement)

Les méthodes dites h-adaptatives consistent à augmenter localement la résolution du maillage par l'ajout de mailles 3.3. Le raffinement, déclenché par le calcul d'un estimateur

d'erreur local, peut être opéré en une seule passe ou de manière récursive (cf. figure 3.2) sur des maillages dont la structure est prévue pour pouvoir gérer un maillage non-structuré.

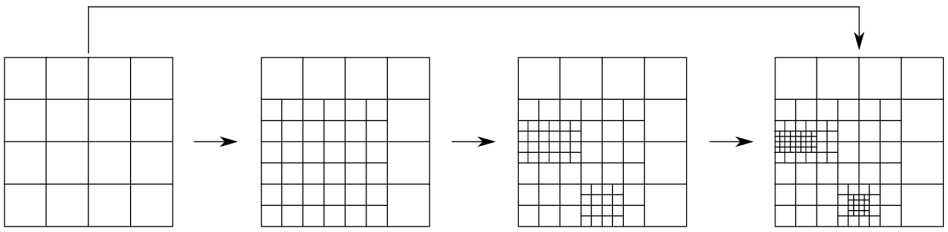


FIGURE 3.2 – Exemple de raffinement h-adaptatif direct ou récursif

De multiples critères de raffinement ont été proposés dans la littérature [143, 74, 78]. Les critères précédents ont été comparés par [40] qui a conclu que celui proposé [74] permet de créer le maillage optimal.

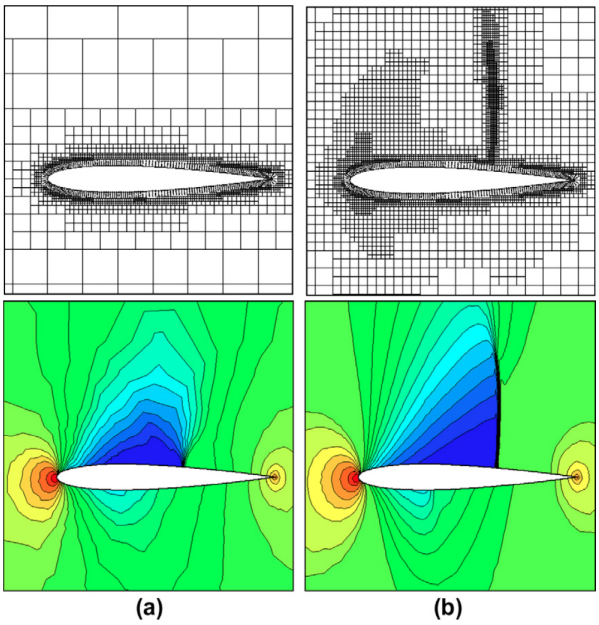


FIGURE 3.3 – Exemple de h-raffinement de maillage de quadrangles : Maillages initial et après quatre niveaux de raffinements d'une aile NACA 0012 et profils des pressions correspondants. issu de [65]

Augmenter le nombre de mailles allonge le temps de calculs. Afin de limiter cet effet des stratégies de déraffinement peuvent être mises en place visant à diminuer le nombre de mailles dans les régions moins sensibles ou non-calculées.

Les méthodes h-adaptatives permettent de gérer des simulations où peuvent apparaître des singularités [6] telles que des ondes de choc dans des écoulements compressibles turbulents ou des fronts dans des écoulements diphasiques [57]. Les méthodes introduites par NAKAHASHI et KIM [94] permettent également de générer des maillages complexes

comme des cavités nasales [80] ou des poumons de rats ou d'humains [38]. Les méthodes h-adaptatives sont les plus répandues que ce soit pour des schémas volume ou éléments finis comme en témoigne la littérature féconde à ce sujet tant en CFD [26, 105] qu'en mécanique des solides [33, 118, 15, 44, 55, 11].

L'application de telles méthodes est utilisable pour des maillages non-structurés. La propriété de conformité du maillage est en général détruite lorsque cette méthode est appliquée sur des maillages cartésiens. Pour conserver la propriété de conformité, pour des maillages initiaux cartésiens il est nécessaire que le solveur puisse gérer plusieurs types de mailles

Une méthode à base de raffinement local de mailles peut également être utilisée pour la génération complète d'un maillage.[38], en remplaçant le critère de raffinement basé sur l'erreur par un critère lié à l'éloignement à la paroi.

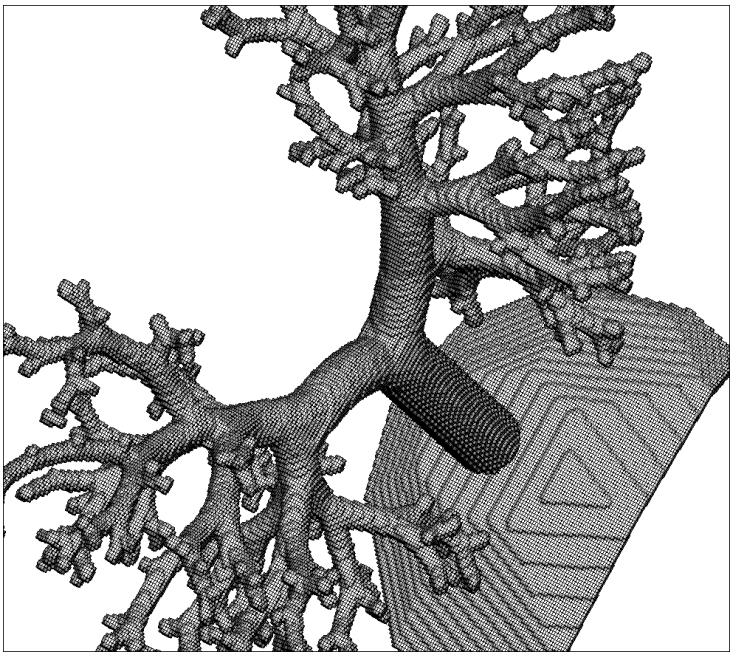


FIGURE 3.4 – Exemple de génération h-adaptative automatique de maillage complexe : un poumon humain [38]

3.1.2 Déformation de Maillage (r-raffinement)

Les méthodes à base de déformation du maillage initial ou r-adaptatives, consistent simplement à déplacer les nœuds du maillage de manière à augmenter la densité de celui-ci dans les régions d'intérêt.

Ces types de méthodes sont utiles pour l'étude de problèmes transitoires où les nœuds du maillage peuvent ainsi suivre les zones où les phénomènes se déplacent.

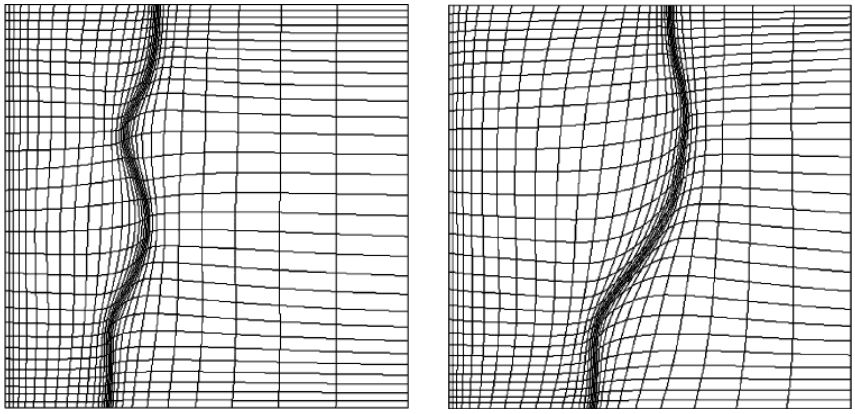


FIGURE 3.5 – Exemple de déformation de maillage avec une méthode r-adaptative pour le suivi des champs de température d'un problème de *Stefan* extrait de [123]

.

On retrouve de nombreux exemples d'utilisation de méthodes r-adaptatives dans la littérature utilisant divers algorithmes pour le déplacement des nœuds [52, 123]. Les méthodes r-adaptatives sont bien adaptées pour les simulations d'écoulements non visqueux compressibles sur des maillages non-structurés ([96, 97]). Les méthodes r-adaptatives permettent de préserver la conformité ainsi que la structuration du maillage et sont relativement faciles en mettre en œuvre. On notera qu'aucun nœud ou maille n'étant ajouté, l'amélioration de la solution est conditionnée par la définition initiale du maillage. De plus cette méthode peut conduire à de forte déformations du maillage rendant le maillage fortement non-orthogonal et donc non propice à un bon traitement des termes visqueux [102].

3.1.3 Augmentation de l'ordre d'approximation de la solution (p-raffinement)

Les méthodes dites p-adaptatives conservent le maillage initial et améliorent la solution en augmentant l'ordre du schéma au voisinage des zones de forte erreur. Ces méthodes ont été développées dans les années 80 par BASU et PEANO [13], BABUSKA et SZABO [6] et BABUSKA, SZABO et KATZ [7]. Contrairement aux autres techniques nécessitant l'ajout d'éléments de maillage, celles-ci sont donc économes en temps de génération de maillage. Ce point positif est contre balancé par l'absence d'amélioration de la géométrie du problème qui ne permet pas une bonne gestion des singularités ou de chocs dont la discontinuité peut-être amplifiée par les fonctions de reconstruction d'ordre élevé BAKER [9] .

Les méthodes p-adaptatives sont adaptées pour les solveurs éléments finis et sont largement étudiées dans la littérature BABUŠKA et SURI [8], BARROS, PROENÇA et BARCELLOS [12], DÜSTER et RANK [39] et KUBATKO et al. [72]

La vitesse de convergence des méthodes p est généralement plus rapide comparée à celle de méthodes h et peut même être exponentielle Babuska, Szabo et Katz [7]. Cependant le solveur devant pouvoir gérer des éléments de plusieurs ordres, ces méthodes sont peu utilisées industriellement car les programmes professionnels utilisent généralement des fonctions de reconstruction d'ordre 2 au maximum (**CAST3M**, Ladevèze et Pelle [73]).

3.1.4 Raffinement par bloc

Les méthodes de raffinement par blocs se décomposent en deux familles. La première méthode dite arborescente (quadtree en 2d et octree en 3d) décompose le maillage principal en blocs où dans les zones d'intérêt un maillage plus raffiné est défini et remplace le maillage initial. Un historique de création des mailles est conservé de manière à pouvoir déraffiner les zones si besoin. Un exemple de décomposition de maillage par cette méthode est illustrée par la figure 3.7(a)

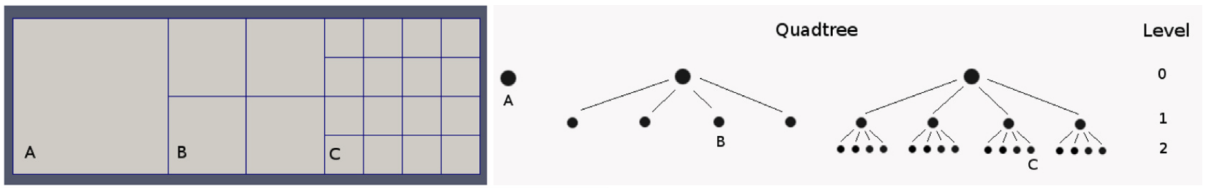
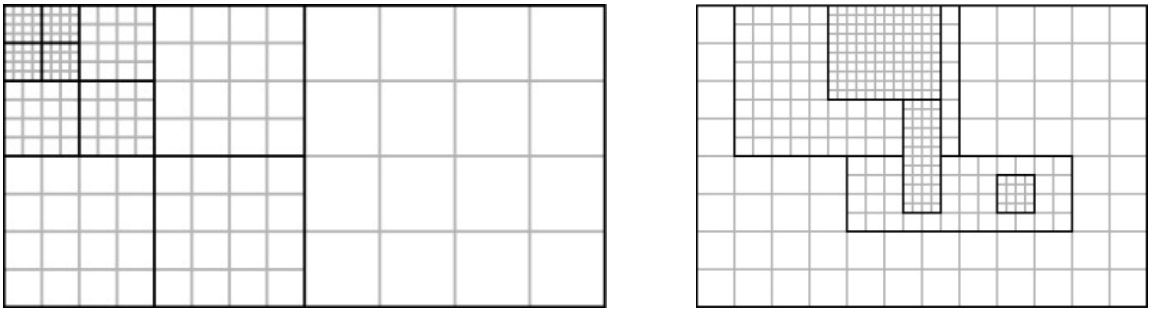


Figure 3.6 – Illustration d'un arbre de parenté quaternaire 2D extrait de Antepara et al. [5]

La seconde méthode dite par patch, définit de nouveau blocs de maillages superposant l'existant et est illustrée par la figure 3.7(b). La méthode que nous proposons dans ce manuscrit s'inscrit donc dans cette catégorie.



(a) Raffinement Arborescent octree (b) Raffinement par patch

Figure 3.7 – Exemples de raffinement par bloc extraits de Zuzio et Estivalezes [144].

Une revue de différents codes utilisés actuellement et librement accessible est proposée dans [37].

3.2 Méthodes multi-grilles locales

Les méthodes multi-grilles locales sont dérivées des méthodes multi-grilles introduites dans les années 70 par Brandt [22]. Ces méthodes accélèrent la convergence. Les hautes fréquences de l'erreur sont rapidement éliminées par un schéma itératif alors que les basses sont éliminées sur les maillages grossiers. Initialement plusieurs maillages recouvrant l'ensemble du domaine sont définis en se basant sur le plus raffiné. Un cycle de prolongements et restrictions lient les différents niveaux de grilles où la solution est résolue jusqu'à convergence sur le maillage le plus détaillé. Les méthodes multi-grilles sont largement étudiées dans la littérature Lubrecht, Ten Napel et Bosma [81], Lebon, Raous et Rosu [77], Wriggers et Boersma [138] et Ghia, Ghia et Shin [47]

Les méthodes multi-grilles locales utilisent donc le même processus d'accélération que les méthodes multi-grilles classiques en restreignant l'utilisation de grilles raffinées au voisinage des zones d'intérêt permettant ainsi un gain de temps de calcul. Les grilles plus raffinées peuvent être ajoutées dynamiquement lors du calcul.

Deuxième partie

Méthodes numériques

Chapitre 4

NSMB - un solveur Navier-Stokes Multi-Blocs

4.1 Historique du Solveur NSMB

En 1989, le solveur structuré EULMB pour **Eul**er **M**ulti-**B**locks est développé au sein de l'École Polytechnique Fédérale de Lausanne (EPFL). Des choix techniques sont à l'origine de l'écriture du solveur :

1. allocation dynamique de la mémoire à l'exécution du programme,
2. parallélisation par résolution partagée de blocs de maillage sur différents processeurs,
3. stockage mémoire dans une base de donnée MEMCOM accélérant les entrées/sorties.

En 1991, l'incorporation de la prise en compte des termes visqueux dans le solveur transforme EULMB en NSMB, **N**avier-**Stokes** **M**ulti-**B**locs. De 1992 à la fin de 2003, le développement du solveur a été maintenu au sein du consortium NSMB, un groupe de travail rassemblant de nombreuses universités EPFL (Lausanne, Suisse) SERAM (Paris, France) IMFT (Toulouse, France) KTH (Stockholm, Suède), Le CERFACS (Toulouse, France), et plusieurs partenaires industriels EADS-France (Airbus France et EADS Space Technologies), SAAB Military Aircraft et CFS Engineering. Depuis 2004, le développement est poursuivi par l'EPF-Lausanne, ETH-Zurich, l'IMFS Strasbourg (devenu le département Mécanique de l'institut ICube), l'IMFT, l'Université militaire de Munich, CFS Engineering et RUAG Aerospace. NSMB reste encore utilisé par Airbus-France, EADS-ST, et KTH.

Le solveur NSMB permet de résoudre les équations de Navier-Stokes, compressibles et incompressibles parla méthode de volumes finis. NSMB a été initialement développé en Fortran 77 et intègre aujourd'hui la syntaxe du Fortran 90 tout en gardant pour le moment la rigidité d'écriture du format fixe. Il permet le parallelisme grâce à la librairie MPI (Message Passing Interface) faisant appel à des maillages découpés en blocs. Le code est très polyvalent et permet un choix de schémas spatiaux et temporels variés, de nombreux modèles de turbulence, la prise en compte de la chimie de l'air en équilibre et hors équilibre, le givrage ou la cavitation. Le manuel de référence [134], détaille l'ensemble

des fonctionnalités du code. Nous reportons par la suite les modèles utilisés pour notre étude.

4.2 Équations de Navier-Stokes

Les écoulements de fluides sont régis par les équations de Navier-Stokes (NS) exprimant la conservation de la masse, le bilan de la quantité de mouvement et d'énergie. Les équations NS compressibles dans leur formulation conservative prennent la forme suivante :

Équation de continuité (ou équation de bilan de masse) :

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \overrightarrow{v}) = 0 \tag{4.1}$$

Équation de bilan de la quantité de mouvement :

$$\frac{\partial \rho \overrightarrow{v}}{\partial t} + \nabla \cdot (\rho \overrightarrow{v} \otimes \overrightarrow{v}) = -\nabla p + \nabla \cdot \overrightarrow{\tau} \tag{4.2}$$

Équation de bilan de l'énergie :

$$\frac{\partial \rho E}{\partial t} + \nabla \cdot [(\rho e + p)\overrightarrow{v})] = \overrightarrow{\nabla} \cdot (\tau \cdot \overrightarrow{v}) - \overrightarrow{\nabla} \cdot \overrightarrow{q} \tag{4.3}$$

avec

- t le temps (en s)
- ρ la masse volumique du fluide (en $kg.m^{-3}$)
- $\boldsymbol{u} = (u, v, w)$ le vecteur vitesse ponctuelle du fluide (en $m.s^{-1}$)
- p la pression (en Pa)
- τ_{ij} le tenseur des contraintes visqueuses (en Pa)
- E l'énergie totale par unité de masse (en $J.kg^{-1}$)
  somme de l'énergie interne E_i et de l'énergie cinétique $E_c = \frac{1}{2}\boldsymbol{u}^2$: $E = E_i + E_c$
- $\overrightarrow{q}$ le flux de chaleur perdu par conduction thermique (en $J.m^{-2}.s^{-1}$)

NSMB utilise le vecteur d'état W définit comme :

$$W = (\rho, \rho u, \rho v, \rho w, \rho E)^T \tag{4.4}$$

qui permet l'écriture des équations NS compressibles stationnaires dans un repère cartésien (x, y, z) sous la forme :

$$\frac{\partial W}{\partial t} + \frac{\partial}{\partial x}(f - f_v) + \frac{\partial}{\partial y}(g - g_v) + \frac{\partial}{\partial z}(h - h_v) = 0 \tag{4.5}$$

où f, g et h sont les flux conservatifs définis par :

$$f = (\rho u \quad ,\rho u^2 + p \quad ,\rho uv \quad ,\rho uw \quad ,u(\rho E + p)\ )^T \tag{4.6}$$
$$g = (\rho v \quad ,\rho vu \quad ,\rho v^2 + p \quad ,\rho vw \quad ,v(\rho E + p))^T \tag{4.7}$$
$$h = (\rho w \quad ,\rho wu \quad ,\rho wv \quad ,\rho w^2 + p \quad ,w(\rho E + p))^T \tag{4.8}$$

et ceux visqueux par :

$$f_v = (0, \tau_{xx}, \tau_{xy}, \tau_{xz}, (\tau U)_x - q_x)^T \tag{4.9}$$
$$f_g = (0, \tau_{yx}, \tau_{yy}, \tau_{yz}, (\tau U)_y - q_y)^T \tag{4.10}$$
$$f_h = (0, \tau_{zx}, \tau_{zy}, \tau_{zz}, (\tau U)_z - q_z)^T \tag{4.11}$$

Le fluide est supposé avoir un comportement newtonien, la partie visqueuse du tenseur des contraintes ne dépend que du tenseur des vitesses de déformation, et ceci de façon linéaire et isotrope :

$$\tau_{ij} = \mu\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right) + \lambda\frac{\partial U_k}{\partial x_k}\delta_{ij} = 2\mu S + \lambda\nabla\cdot\boldsymbol{u} \tag{4.12}$$

où λ et μ sont les coefficients de Lamé représentant respectivement la viscosité dynamique et le second coefficient de viscosité du fluide. L'hypothèse de Stokes supposant l'égalité entre pression mécanique et thermodynamique permet d'écrire :

$$2\mu + 3\lambda = 0. \tag{4.13}$$

Les termes du tenseur des contraintes visqueuses se développent sous la forme :

$$\tau_{xx} = \frac{2}{3}\left(2\frac{\partial u}{\partial x} - \frac{\partial v}{\partial y} - \frac{\partial w}{\partial z}\right), \qquad \tau_{xy} = \tau_{yx} = \mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) \tag{4.14}$$
$$\tau_{yy} = \frac{2}{3}\left(-\frac{\partial u}{\partial x} + 2\frac{\partial v}{\partial y} - \frac{\partial w}{\partial z}\right), \qquad \tau_{yz} = \tau_{zy} = \mu\left(\frac{\partial v}{\partial z} + \frac{\partial z}{\partial y}\right) \tag{4.15}$$
$$\tau_{zz} = \frac{2}{3}\left(-\frac{\partial u}{\partial x} - \frac{\partial v}{\partial y} + 2\frac{\partial w}{\partial z}\right), \qquad \tau_{zx} = \tau_{xz} = \mu\left(\frac{\partial w}{\partial x} + \frac{\partial u}{\partial z}\right) \tag{4.16}$$

La dissipation visqueuse de l'équation de l'énergie est alors obtenue par :

$$(\tau U)_x = \tau_{xx}u + \tau_{xy}v + \tau_{xz}w \tag{4.17}$$
$$(\tau U)_y = \tau_{yx}u + \tau_{yy}v + \tau_{yz}w \tag{4.18}$$
$$(\tau U)_z = \tau_{zx}u + \tau_{zy}v + \tau_{zz}w \tag{4.19}$$

Le mode de transmission de chaleur ést considéré comme uniquement conductif, le schéma de conduction thermique suit alors la loi de Fourier, liant proportionnellement le vecteur de flux de chaleur au gradient de température :

$$q_x = -k\frac{\partial T}{\partial x}, q_y = -k\frac{\partial T}{\partial y}, q_z = -k\frac{\partial T}{\partial z} \tag{4.20}$$

avec la température T et k le coefficient de conductivité thermique. La viscosité laminaire μ est donnée par la loi de Shuterland :

$$\frac{\mu}{\mu_\infty} = \left(\frac{T}{T_\infty}\right)^{3/2} \frac{(T_\infty + S_1)}{(T + S_1)} \tag{4.21}$$

μ_∞ étant la viscosité du fluide établie à la température T_0 et S est une constante égale à 110.3 K pour l'air. La valeur du coefficient k est déduite en supposant le nombre de Prandtl constant pour l'air ($Pr = 0.72$). Ainsi on vérifie que

$$k = \frac{\mu C_p}{Pr} \qquad \text{avec} \quad C_p = \gamma C_v \qquad \text{et} \;\; C_v = \frac{R}{\gamma - 1} \tag{4.22}$$

C_p et C_v étant respectivement les coefficients de chaleur spécifique à pression constante et à volume constant, $R = 287 JKg^{-1}K^{-1}$ est la constante des gaz parfaits et $\gamma = 1.4$ le rapport des coefficients de chaleur spécifique. L'équation des gaz parfaits lie la pression aux autres variables thermodynamiques :

$$p = \rho RT, \qquad E_i = C_v T = \frac{1}{\gamma - 1}\frac{p}{\rho}. \tag{4.23}$$

L'énergie totale est reliée à l'énergie interne E_i par

$$E = E_i + \frac{1}{2}\boldsymbol{u}^2 \tag{4.24}$$

4.2.1 Schéma de résolution temporelle

La discrétisation temporelle s'appuie sur une méthode implicite à pas de temps dual et inversion des matrices par une méthode LU-SGS proposée par Weber [136]. La discrétisation volumes finis mène au système d'équations différentielles :

$$\frac{d}{dt}\left(V_{(i,j,k)} W_{(i,j,k)}\right) + Q_{(i,j,k)} - D_{(i,j,k)} = 0 \tag{4.25}$$

où W est le vecteur d'état localisé au centre d'une cellule (i, j, k), V le volume et Q le flux entrant et sortant de la cellule. Soit l'équation :

$$\frac{d}{dt}\left(V_{(i,j,k)} W_{(i,j,k)}\right) + R_{(i,j,k)} = 0 \tag{4.26}$$

où le résidu $R_{(i,j,k)} = Q_{(i,j,k)} - D_{(i,j,k)}$, est la somme des flux orientés entrants et sortants Q et du terme de dissipation D sur une cellule (i, j, k). Pour une cellule dont le volume reste fixe au cours du temps, le terme volumique peut être extrait de la dérivée temporelle :

$$V_{(i,j,k)} \frac{d}{dt} \left(W_{(i,j,k)} \right) + R_{(i,j,k)} = 0. \tag{4.27}$$

De multiples schémas de discrétisation temporelle peuvent être appliqués au système 4.27. Les schémas explicites étant trop contraignants à l'utilisation sur des maillages finement raffinés, des schémas implicites d'ordre plus élevé sont introduits. Nous utiliserons pour les simulations des schémas d'intégration itératifs linéaires où l'évolution temporelle d'une variable est exprimée sous la forme d'une combinaison linéaire de valeurs de cette variable prises à des instants différents. La méthode linéaire à deux pas introduite par Magarvey et Bishop [82] est décrite ici pour des pas de temps variables :

$$(1+\xi)\frac{\Delta W^n}{\Delta t^n} V - \xi \frac{\Delta W^{n-1}}{\Delta t^{n-1}} V = -\left(\theta R^{n+1} + (1 - \theta + \phi) R^n - \phi R^{n-1} \right) \tag{4.28}$$

où $\Delta W^n = W^{n+1} - W^n$ et Δt_n l'incrément de temps séparant les instants n et $n+1$. Le schéma est caractérisé par les coefficients θ, ξ et $\phi \in [0, 1]$. Une valeur $\theta = 0$ implique la disparition du terme résiduel implicite R^{n+1} et définit donc un schéma explicite. Inversement pour toute valeur non nulle de θ le schéma est alors implicite. Le schéma implicite aux différences rétrogrades (*Backward differentiation*) correspondant aux paramètres $\theta = 1$, $\xi = 1/2$ et $\phi = 0$ est choisi pour l'ensemble de cette thèse.

4.3 Méthode du pas de temps dual "Dual Time Stepping"

Une fois la discrétisation temporelle de l'équation (4.28) appliquée, le système non linéaire d'équations reste à être résolu pour chaque incrément de temps :

$$H\left(W^{n+1}\right) = (1+\xi)\frac{\Delta W^n}{\Delta t^n} V - \xi \frac{\Delta W^{n-1}}{\Delta t^{n-1}} V + \theta R^{n+1} + (1 - \theta + \phi) R^n - \phi R^{n-1} = 0 \tag{4.29}$$

Pour chaque pas de temps externe, le résidu $H(U)$ est amené à converger vers 0 via une méthode itérative. L'amorce de l'approche par pas de temps dual passe par l'utilisation du résidu instationnaire H dans le système d'équations différentielles suivant :

$$V \frac{dW}{dt^*} + H(W) = 0 \tag{4.30}$$

où t^* est un temps fictif. Le système est alors intégré jusqu'à l'obtention d'un état station-

naire. Le problème revient donc à résoudre un système d'équations différentielles ordinaires à chaque pas de temps physique Δt. Les méthodes explicites ou implicites développées pour la résolution de problèmes stationnaires peuvent donc être choisies pour la résolution ce système.

La discrétisation de l'équation (4.30) pour le temps fictif par un schéma décentré d'Euler conduit à la formulation :

$$V\frac{\Delta W^{\nu}}{\Delta t^{*}} + H(W^{n+1,\nu+1}) = 0 \tag{4.31}$$

où ν désigne l'indice des itérations internes et $\Delta W^{\nu} = W^{n+1,\nu+1} - W^{n+1,\nu}$. La linéarisation du résidu instationnaire autour des instants fictifs ν donne l'équation :

$$H\left(W^{n+1,\nu+1}\right) = H\left(W^{n+1,\nu}\right) + \left.\frac{\partial H\left(W^{n+1}\right)}{\partial W^{n+1}}\right|^{\nu} \Delta W^{\nu} + O(\Delta t^{*2}) \tag{4.32}$$

soit en remplaçant $H\left(W^{n+1,\nu+1}\right)$ par son développement précédent dans l'équation (4.31), le système linéaire suivant peut être obtenu :

$$\left(\frac{V}{\Delta t^{*}}I + \left.\frac{\partial H\left(W^{n+1}\right)}{\partial W^{n+1}}\right|^{\nu}\right)\Delta W^{\nu} = -H\left(W^{n+1,\nu}\right) \tag{4.33}$$

qui peut être développé plus avant en écrivant la linéarisation du résidu en :

$$\left(\frac{V}{\Delta t^{*}}I + (1+\xi)\frac{V}{\Delta t}I + \theta\left.\frac{\partial R}{\partial U}\right|^{\nu}\right)\Delta U^{\nu} = -H\left(U^{n+1,\nu}\right) \tag{4.34}$$

Ce dernier système peut être résolu avec une méthode LU-SGS (*Lower-Upper Symmetric Gauss-Seidel*, Weber [136]. L'extension de la méthode LU-SGS au cas stationnaire se résume à inclure le pas de temps fictif, les facteurs $(1+\xi)$ et θ au sein de l'opérateur implicite ainsi qu'à ajouter les termes sources dans au second membre de l'équation.

4.4 Discrétisation spatiale : schéma centré et dissipation artificielle

Dans nos différentes simulations nous utilisons le schéma centré d'ordre 4 de [66] qui se développe dans une direction i sous la forme :

$$W_{i+\frac{1}{2},j,k} = \frac{W_{i+2,j,k} + 7\times W_{i+1,j,k} + 7\times W_{i-1,j,k} - W_{i-2,j,k}}{6} \tag{4.35}$$

Le schéma centré doit être augmenté d'un terme de viscosité artificielle du second ordre au voisinage de discontinuités ainsi que d'un terme de dissipation du quatrième ordre destiné à supprimer les oscillations paires / impaires. Le terme de dissipation est similaire à ceux de dissipation visqueuse. Il fait appel aux dérivées d'ordre deux et quatre

du vecteur d'état, dérivées qui sont alors pondérées à l'aide d'un facteur d'échelle et d'un poids. Ce dernier poids est construit avec la valeur absolue de la dérivée seconde normalisée de la pression. Par construction, ce poids est donc faible sur l'ensemble du domaine excepté dans les zones de fort gradient de pression que l'on retrouve classiquement au voisinage des zones de choc ou des points de stagnation. Il constitue donc un "switch" permettant d'activer effectivement le terme de dissipation visqueuse dans les zones précédemment évoquées pouvant générer des oscillations. Le terme de dissipation du quatrième ordre est quant à lui utilisé partout à l'exception des zones où la dissipation du second ordre est active et ceci afin d'éviter la production d'oscillations dans ces dernières régions. Après addition des termes de dissipation le schéma numérique s'écrit :

$$\frac{d}{dt}\left(V_{i,j,k}W_{i,j,k}\right) + Q_{i,j,k} - D_{i,j,k} = 0 \tag{4.36}$$

où $Q_{i,j,k}$ correspond à la discrétisation de l'équation de quantité de mouvement et $D_{i,j,k}$ à celle de la dissipation artificielle. L'opérateur D est décomposé comme suit :

$$D_{i,j,k} = \left(d_{i+1/2,j,k} - d_{i-1/2,j,k}\right) + \left(d_{i,j+1/2,k} - d_{i,j-1/2,k}\right) + \left(d_{i,j,k+1/2} - d_{i,j,k-1/2}\right) \tag{4.37}$$

où chaque flux dissipatif est évalué aux faces. Le flux $d_{i+1/2,j,k}$ dans la direction i est calculé à partir de la relation :

$$\begin{aligned} d_{i-1/2,j,k} =& r_{i-1/2,j,k}\left[\epsilon^{(2)}_{i-1/2,j,k}\left(W_{i,j,k} - W_{i-1,j,k}\right)\right. \\ & \left. -\epsilon^{(4)}_{i-1/2,j,k}\left(W_{i+1,j,k} - 3W_{i,j,k} + 3W_{i-1,j,k} - W_{i-2,j,k}\right)\right] \end{aligned} \tag{4.38}$$

et des expressions analogues pour les directions j et k sont utilisées. Dans l'équation (4.38) r est un facteur d'échelle reliant le flux dissipatif à l'amplitude des flux convectifs à travers cette face. Les coefficients $\epsilon^{(2)}$ et $\epsilon^{(4)}$ permettent d'adapter localement les flux dissipatifs. Comme mentionné précédemment, la dissipation du deuxième ordre est directement liée à la normalisation de la dérivée du second ordre de la pression. Pour la direction i ce terme peut être écrit comme :

$$\mu_{i,j,k} = \left|\frac{p_{i+1,j,k} - 2p_{i,j,k} + p_{i-1,j,k}}{p_{i+1,j,k} + 2p_{i,j,k} + p_{i-1,j,k}}\right| \tag{4.39}$$

Le terme switch s'écrit

$$\nu_{i-1/2,j,k} = max\left(\mu_{i-1,j,k}, \mu_{i,j,k}\right). \tag{4.40}$$

Le tenseur ν prend donc la plus grande valeur de la dérivée seconde normalisée de part

et d'autre de la face $i+1/2$. Les coefficients $\epsilon^{(2)}$ et $\epsilon^{(4)}$ sont définis comme :

$$\begin{aligned} \epsilon^{(2)}_{i-1/2,j,k} &= k^{(2)}\nu_{i-1/2,j,k} \\ \epsilon^{(4)}_{i-1/2,j,k} &= max\left(0, k^{(4)} - \epsilon^{(2)}_{i-1/2,j,k}\right) \end{aligned} \tag{4.41}$$

De cette manière le terme de dissipation du $4^{ème}$ ordre est automatiquement désactivé au voisinage de toute discontinuité. Des expressions similaires peuvent être écrites pour les directions j et k. Les paramètres $k^{(2)}$ et $k^{(4)}$ sont des constantes utilisées pour contrôler la dissipation. Les valeurs typiques en incompressible sont $k^{(2)} = 0$ et $k^{(2)} = 1/64$.

4.5 Conditions Limites

Deux rangées de cellules dites fictives sont ajoutées sur le pourtour du maillage afin de gérer les conditions limites. Les valeurs imposées à ces cellules dépendent de la nature de la condition limite. Les vitesses des cellules fictives des parois sont fixées en fonction de la nature du mouvement. On considérera des condtions de type de Dirichlet : $U_{paroi} = 0$ ou $U_{paroi} = U_{déplacement}$. Pour la pression ainsi que la densité, un gradient nul est imposé.

$$\frac{\partial}{\partial t}(W) + \frac{\partial}{\partial x}f + \frac{\partial}{\partial y}g + \frac{\partial}{\partial z}h = 0. \tag{4.42}$$

Les valeurs propres des équations d'Euler sont aisément obtenues via la forme non-conservative des équations en fonction des variables primitives $V = (\rho, u, v, w, p)^T$. La transformation vers la formulation non-conservatrice des variables s'écrit :

$$\frac{\partial}{\partial t}V + A\frac{\partial}{\partial x}V + B\frac{\partial}{\partial y}V + C\frac{\partial}{\partial z}V = S \tag{4.43}$$

où A, B et C représentent les composantes de la matrice jacobienne $\overrightarrow{A}$, et S représente le terme source. Les valeurs propres de cette dernière matrice jacobienne sont obtenues en résolvant le système

$$det\left|\lambda\overline{\overline{I}} - \overrightarrow{A}\cdot\overrightarrow{n}\right| = 0 \tag{4.44}$$

où $\overrightarrow{n}$ correspond à la direction de propagation des ondes. Pour un gaz parfait en 3D, on peut montrer que la matrice Jacobienne $\overrightarrow{A}\cdot\overrightarrow{n}$ possède cinq valeurs propres lorsque aucune autre advection scalaire n'est prise en compte. Ces valeurs s'écrivent :

$$\lambda_1 = \lambda_2 = \lambda_3 = \overrightarrow{v}\cdot\overrightarrow{n} \tag{4.45}$$

$$\lambda_4 = \overrightarrow{v}\cdot\overrightarrow{n} + c \qquad \lambda_5 = \overrightarrow{v}\cdot\overrightarrow{n} - c \tag{4.46}$$

où c représente la vitesse du son. Pour chaque valeur propre λ_i, le vecteur propre associé l_i peut être trouvé. De plus, pour chaque valeur propre une relation de compatibilité peut-être déduite sous la forme :

$$l_i \frac{\partial}{\partial t} V + l_i \left(\overrightarrow{A} \cdot \overrightarrow{\nabla} \right) V = S \tag{4.47}$$

Les cinq relations de compatibilités peuvent être factorisées sous une forme plus compacte en introduisant L^{-1} la matrice composée des vecteurs propres à gauche l_i de la matrice L telle que $L^{-1}L = I$,

$$\left(L^{-1} \frac{\partial}{\partial t} + L^{-1} \left(\overrightarrow{A} \cdot \overrightarrow{\nabla} \right) \right) V = S. \tag{4.48}$$

L^{-1} étant composée des vecteurs propres à gauche, on peut écrire que :

$$L^{-1} \overrightarrow{A} \cdot \overrightarrow{n} = \Lambda L^{-1} \tag{4.49}$$

où Λ est la matrice diagonale de l'ensemble des valeurs propres. En utilisant un nouvel ensemble de variables caractéristiques associées à une direction de propagation donnée $\overrightarrow{n}$ il est possible de transformer l'équation (4.48) comme

$$L^{-1} \frac{\partial V}{\partial t} + \left(L^{-1} \overrightarrow{A} L \right) L^{-1} \overrightarrow{\nabla} V = S \tag{4.50}$$

soit

$$L^{-1} \frac{\partial V}{\partial t} + \Lambda L^{-1} \overrightarrow{\nabla} V = S. \tag{4.51}$$

L'équation (4.51) est utilisée pour calculer les variables dans les cellules fictives selon le signe de la valeur propre λ_i. La forme discrétisée s'écrit :

$$(l_i)^n_{ghost} V^{n+1}_{ghost} = \begin{cases} (l_i)^n_{ghost} V^{n+1}_{ghost} & \text{si } \lambda_i < 0 \\ \\ (l_i)^n_{ghost} V^{n+1}_{intérieur} & \text{si } \lambda_i > 0 \end{cases} \tag{4.52}$$

Cette méthode de proise en compte des conditions aux limites est obtenue pour des flux non-visqueux et peut être également appliquée à des écoulements visqueux.

4.6 Préconditionneur pour l'incompressibilité

Les écoulements présents dans cette étude sont incompressibles. Afin de satisfaire la condition d'incompressibilité l'usage est fait d'une technique de préconditionnement. Le préconditionnement implémenté dans le solveur NSMB et utilisé lors de ce travail repose sur la méthode de compressibilité artificielle proposée par Chorin [29]. Avec ce

préconditionnement le vecteur d'état est transformé en :

$$W = (p \quad ,\rho u \quad ,\rho v \quad ,\rho w \quad)^T$$

$$\begin{aligned} f &= (\rho u \quad ,\rho u^2 + p \quad ,\rho uv \quad ,\rho uw \quad)^T \\ g &= (\rho v \quad ,\rho vu \quad ,\rho v^2 + p \quad ,\rho vw \quad)^T \\ h &= (\rho w \quad ,\rho wu \quad ,\rho wv \quad ,\rho w^2 + p)^T \end{aligned}$$

et l'équation du système à résoudre est réécrite sous la forme :

$$P^{-1}\frac{\partial}{\partial \tau}(W) + \overline{I}\frac{\partial}{\partial t}(W) + \frac{\partial}{\partial x}(f - f_v) + \frac{\partial}{\partial y}(g - g_v) + \frac{\partial}{\partial z}(h - h_v) = 0. \tag{4.53}$$

La formulation des flux conservatifs et visqueux restent respectivement identiques aux formulations 4.8 et 4.11. La matrice identité modifiée $\overline{I}$ et la matrice de préconditionnement P pour le système d'équations pseudo-temporel est donnée par :

$$\overline{I} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{et} \quad P = \begin{bmatrix} \beta^2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.54}$$

avec le coefficient β pouvant être considéré comme un paramètre de relaxation pour la solution pseudo-temporelle défini comme

$$\beta^2 = max\left(\beta_{min}^2 U_\infty^2,\ C_\beta U^2\right) \tag{4.55}$$

où $\beta_{min} = 5$ et $C_\beta = 0,05$.

4.7 Formulation ALE

Les mouvements de solides ou les déformations de structures se traduisent respectivement par des déplacements, ou déformations de maillages. Ces modifications du maillage nécessitent une modification des équations pour rendre compte des déplacements des nœuds du maillage. La formulation ALE pour *Arbitrary Lagrangian Eulerian*, est donc introduite au sein des équations NS à cet effet. Le maillage est alors considéré comme mobile à une vitesse $\overrightarrow{W}$. Cette vitesse permet d'appliquer arbitrairement un point de vue totalement Lagrangien ou Eulérien respectivement en égalisant cette vitesse à celle du fluide par exemple au voisinage d'une paroi pour une bonne résolution de l'interface fluide-structure, ou en l'égalisant à une valeur nulle à l'intérieur du domaine de calcul. La transition entre le point de vue purement Lagrangien ou Eulérien peut s'effectuer de

manière continue, ceci permettant d'allier les avantages des deux points de vue. Dans nos simulations les déformations de maillage ne sont pas prises en considération, seuls les déplacements de grilles le sont. Dans ce cas de figure, les équations i(4.5) peuvent être reformulées :

$$\frac{\partial W}{\partial t} + \frac{\partial}{\partial x}(f - f_v) + \frac{\partial}{\partial y}(g - g_v) + \frac{\partial}{\partial z}(h - h_v) = 0 \tag{4.56}$$

les flux convectifs f, g et h étant définis suivant les directions x,y et z par :

$$f = (\rho\tilde{u} \quad ,\rho u\tilde{u} + p \quad ,\rho u\tilde{v} \quad ,\rho u\tilde{w} \quad)^T \tag{4.57}$$

$$g = (\rho\tilde{v} \quad ,\rho v\tilde{u} \quad ,\rho v\tilde{v} + p \quad ,\rho v\tilde{w} \quad)^T \tag{4.58}$$

$$h = (\rho\tilde{w} \quad ,\rho w\tilde{u} \quad ,\rho w\tilde{v} \quad ,\rho w\tilde{w} + p)^T \tag{4.59}$$

avec chaque composante de vitesse tildée étant la différence entre la composante de vitesse du fluide et celle du maillage $\tilde{u} = u - u_{maillage}$, $\tilde{v} = v - v_{maillage}$, $\tilde{w} = w - w_{maillage}$. Les flux visqueux restent identiques à l'équation (4.11).

Chapitre 5

Méthode Chimère

Ce chapitre est consacré à la méthode Chimère, méthode numérique permettant la gestion de maillages superposés. Une première partie présente une description générale de la méthode. Une seconde décrit les variantes proposées dans la littérature.

5.1 Principe de fonctionnement de la méthode Chimère

Le principe de la méthode chimère peut être résumé par le concept de "*Diviser pour mieux mailler*". La méthode consiste à définir un recouvrement géométrique du domaine de calcul par des maillages indépendants et non disjoints. de la géométrie globale d'un problème. L'échange d'information s'opère au niveau des zones de superposition.

Prenons l'exemple plan d'un cylindre immergé et immobile dans une conduite rectiligne. Avec la méthode chimère, un premier maillage est créé pour la conduite et un second pour le cylindre. Ces deux géométries ne nécessitent aucun effort particulier de génération.

L'exemple précédent malgré sa simplicité nécessite une certaine attention lors de la création d'un maillage purement structuré sans méthode chimère, attention qui doit être

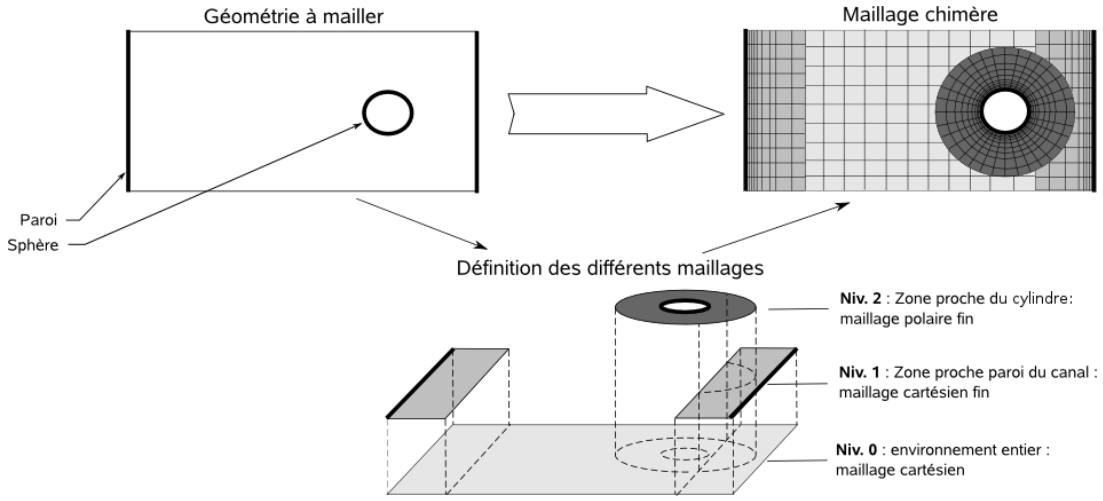


Figure 5.1 – Schéma de principe de la méthode chimère

renouvelée si de multiples positions sont à étudier. La méthode chimère simplifie donc grandement le processus de création de maillages et permet la gestion de maillages mobiles sans déformation de maillage potentiellement nuisible à la qualité des calculs.

Les pionniers de la méthode chimère sont Benek, Steger et Dougherty [16] en 1983 . Par la suite la méthode a été reprise et affinée au cours du temps pour résoudre de nombreux problèmes. La littérature est riche d'applications : étude du débit sanguin dans les poumons, écoulements à vitesse élevée [59, 60, 69], écoulements sanguin dans un cœur artificiel [71], combustion [127], aéronautique [24, 98, 84, 67, 85], écoulement autour de navires [119], séparation des boosters d'une navette spatiale [86] ou étude acoustique [34] démontrent par l'usage la grande souplesse et l'adaptabilité de la méthode.

Pour illustrer la méthode, les maillages utilisés par Tang, Casey Jones et Sotiropoulos [122] pour simuler l'écoulement dans un canal avec deux obstacles, par Meakin et Suhs [86] pour discrétiser une navette spatiale et son lanceur ou encore par Fujii [46] pour simuler l'écoulement autour d'un train entrant dans un tunnel sont représentés figure 5.2.

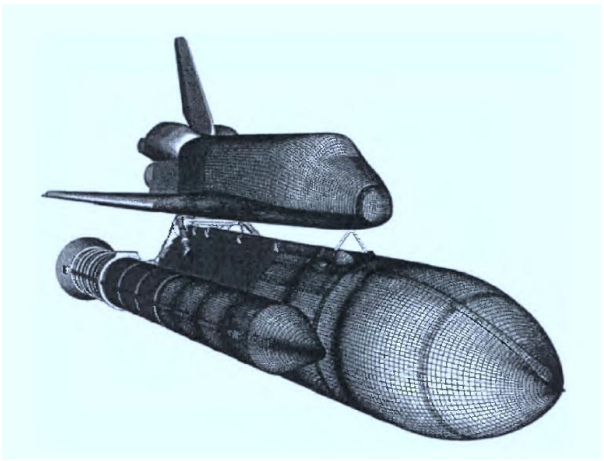
La méthode chimère détaillée pour l'application aux maillages structurés est décrite par Meakin [85]. La méthode est inspirée par la définition de Landmann [76] et est divisée en deux phases et quatre étapes principales et a été implantée dans le code NSMB par DELOZE [31] :

- INITIALISATION CHIMÈRE

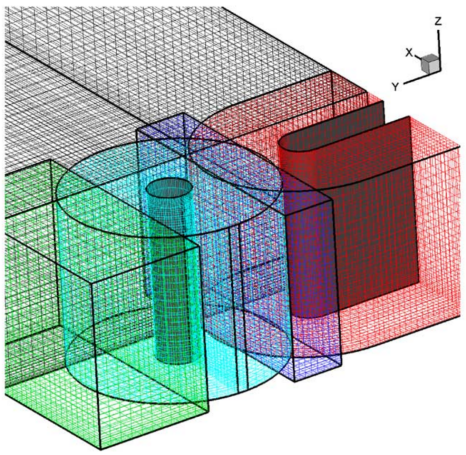
---

1. **Détecter les cellules superposées**. Test basé sur les coordonnées des cellules
2. **Catégoriser les cellules superposées** · selon les 3 types [76] :
   - **Interpolée** à partir des informations des cellules la superposant
   - **Calculée** à partir des équations caractéristiques
   - **Ignorée**. Quelles soit blanchie ou masquée ce type de cellule n'intervient pas dans la résolution numérique du problème, ni dans les interpolations.
3. **Calculer les paramètres d'interpolation** i.e. associer les cellules calculées aux cellules interpolées et définir les poids d'interpolation des cellules calculées.

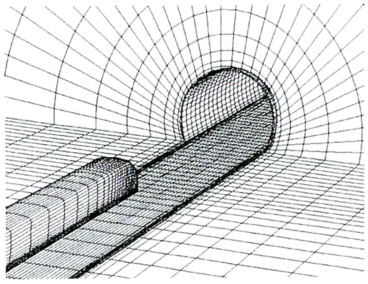
- COMMUNICATION CHIMÈRE

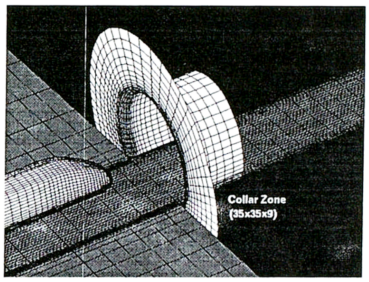


(a)



(b)





(c)

FIGURE 5.2 – Maillage utilisé par MEAKIN et SUHS [86] pour simuler la séparation de l'orbiteur de la navette spatiale des propulseurs principaux (extrait de MEAKIN [85]) (a), maillage chimère utilisée par TANG, CASEY JONES et SOTIROPOULOS [122] pour simuler l'écoulement dans un canal avec deux obstacles (extrait de [122]) (b), maillages utilisés par FUJII [46] pour simuler l'écoulement autour d'un train entrant dans un tunnel (cf. FUJII [46]) (c).

4. ***Calculer les valeurs interpolées*** i.e. utiliser les poids pour définir les valeurs des cellules interpolées à partir des cellules calculées.

La procédure précédente comporte deux phases principales. La première phase est l'initialisation chimère au cours de laquelle les superpositions de cellules sont détectées et les communications déterminées. La seconde est la communication chimère à proprement parler où les informations entre cellules calculées et interpolées sont échangées. Pour des grilles statiques la première phase est opérée seulement en début de calcul alors que pour des grilles mobiles l'initialisation doit être répétée à chaque pas de temps car les superpositions sont évidemment modifiées.

5.2 Détail des étapes de la méthode chimère

5.2.1 Détecter les cellules superposées

La détection des cellules superposées est la première étape. L'organisation des cellules superposées est en général aléatoire, c'est pourquoi il est nécessaire de procéder à la détection en s'appuyant sur les coordonnées des cellules. Cette détection est accélérée par une méthode de grille virtuelle cartésienne uniforme dont la description est disponible dans Meakin [85], Samet [110] et Siikonen, Rautaheimo et Salminen [113].

5.2.2 Catégoriser les cellules superposées

La seconde étape de la méthode chimère consiste à déterminer quelles sont les cellules les plus aptes à être le support du calcul de l'écoulement. De manière générale et intuitive les cellules les plus aptes sont celles possédant la meilleure résolution locale. La détermination des cellules calculées s'opère au travers d'une hiérarchie de critères :

1. La hiérarchie des blocs définie manuellement
2. La qualité chimère des cellules définie manuellement
3. Le volume des cellules comparé automatiquement
4. La distance normale des cellules à la plus proche paroi

Pour des superpositions simples comme vues avec le précédent exemple d'un cylindre fixe dans une canalisation, le critère de hiérarchie de bloc défini préalablement par l'utilisateur est suffisante et permet une catégorisation rapide. Les cellules composant le maillage entourant le cylindre sont définies comme dominantes et seront celles calculées dans la zone de superposition.

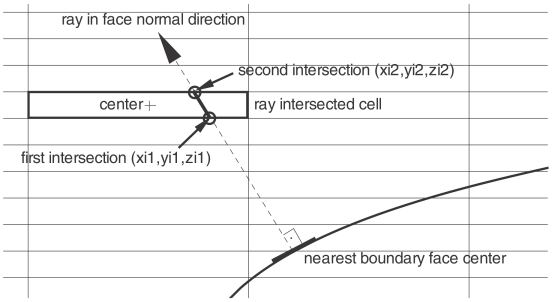


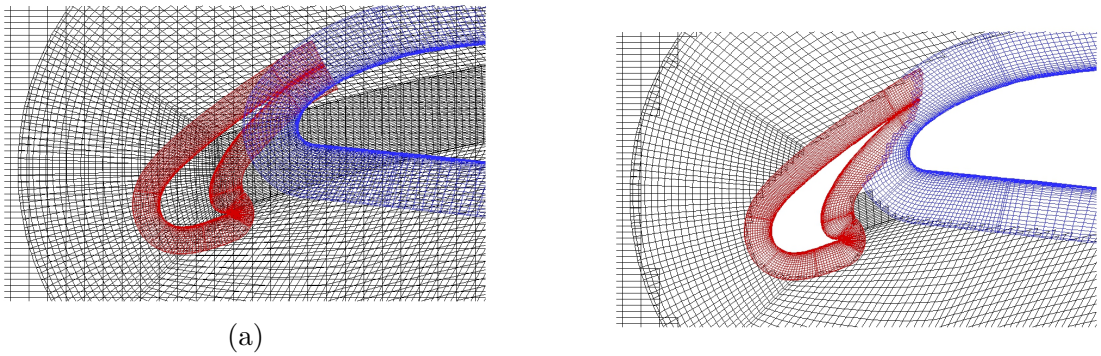
FIGURE 5.3 – Critère de qualité basé sur la distance de l'intersection entre la normale à la paroi et la cellule (extrait de LANDMANN et MONTAGNAC [76]).

Pour des cas plus généraux de superpositions multiples et rapprochées de grilles, le simple critère de hiérarchie des blocs n'est plus suffisant. Les trois critères suivants permettent donc de gérer les cas complexes :

- Le critère de qualité chimère permet à l'utilisateur, pour chaque cellule, de fixer manuellement quelles seront les "meilleures" cellules aptes à être le support du calcul de l'écoulement. Ce critère peut avoir son utilité pour des cas statiques, cependant en cas de mouvements de grilles la nécessité de redéfinir manuellement les cellules rend ce critère peu pratique.
- Le critère basé sur le volume des cellules indique de manière automatique les cellules les plus petites devant être utilisées pour calculer l'écoulement. Les cellules plus petites indiquent en effet une meilleure discrétisation locale du domaine qui assure donc une meilleure résolution de celui-ci SIIKONEN, RAUTAHEIMO et SALMINEN [113] ou LIAO, CAI et TSAI [79].
- Le critère basé sur la distance des cellules dans la direction normale à la plus proche paroi permet de préserver automatiquement une discrétisation fine proche des parois. Ce critère requiert pour chaque cellule, la détermination de la paroi la plus proche suivie du calcul de la norme du segment d'intersection entre la normale à cette paroi avec les limites de la cellule. Distance de sécurité à la paroi manuelle et calcul automatique du volume des cellules permettent finalement de déterminer les cellules à qualifier comme dominantes. Ce critère permet de préserver une qualité physique primordiale à un bon calcul, à savoir les couches limites. Il est cependant coûteux en temps de calcul surtout pour les grilles mobiles car il doit être recalculé à chaque instant.

5.2.3 Calculer les paramètres d'interpolation

La troisième étape de l'initialisation chimère définit la communication entre les blocs. La précision des interpolations nécessaire pour la méthode chimère a fait l'objet de nombreuses études [28, 112]. Cette précision dépend du type d'application physique. Par



(a)

FIGURE 5.4 – Exemple de maillages chimère de bec de bord d'attaque d'une aile d'avion. a) blocs cartésiens, b) blocs curvilignes [32]

exemple il peut être noté que les applications en acoustique requièrent une grande précision dans le calcul des interpolations [86].

De nombreux types d'interpolations sont utilisés pour la méthode chimère. On peut noter les interpolations de type trilinéaire [61, 122, 27, 79] - la plus répandue -, quadratique [25], tétravolumique [120, 113] ou basées sur les polynômes de Lagrange [34, 58, 141, 111, 112]. Ces interpolations peuvent êtres purement explicites - se basant uniquement sur les valeurs de cellules calculées - ou implicites - se basant sur les valeurs de cellules également interpolées. L'interpolation implicite peut être évitée en interdisant la possibilité pour une cellule interpolée d'être elle-même donneuse ou éloignant les zones d'interpolations [112].

La superposition arbitraire des blocs présents dans la méthode chimère induisent une perte de la propriété de conservation de masse [135]. Des travaux ont été menés pour assurer la conservation du flux pour les écoulements compressibles [106] ou incompressibles [139].

Deux exemples de résultats de simulations d'écoulements turbulents compressibles obtenus avec les maillages dont le détail apparait dans la figure 5.4 sont reproduits en figures 5.5 et 5.6.

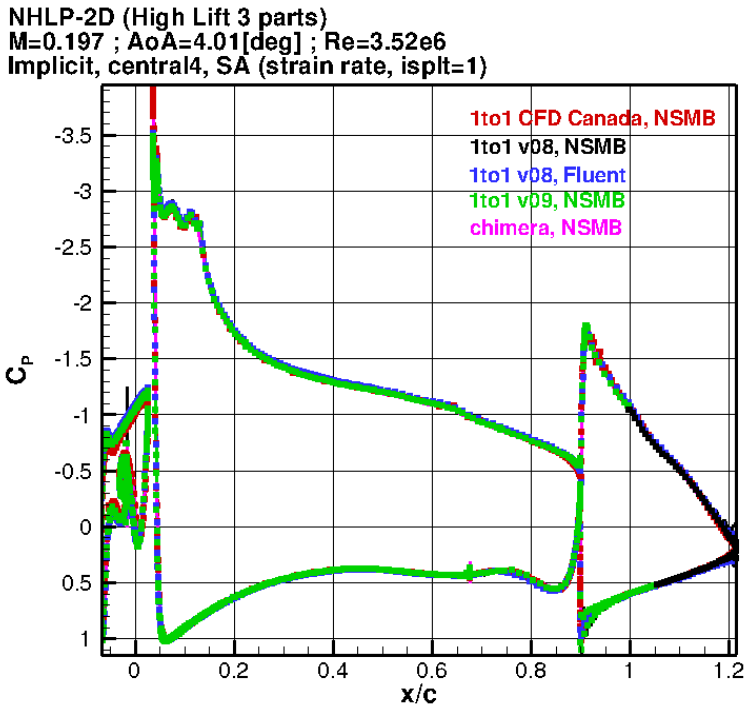


FIGURE 5.5 – Comparaison des coefficients de pression obtenus par NSMB et Fluent sur maillage non-chimère et chimère d'un profil 3 éléments High-Lift [32]

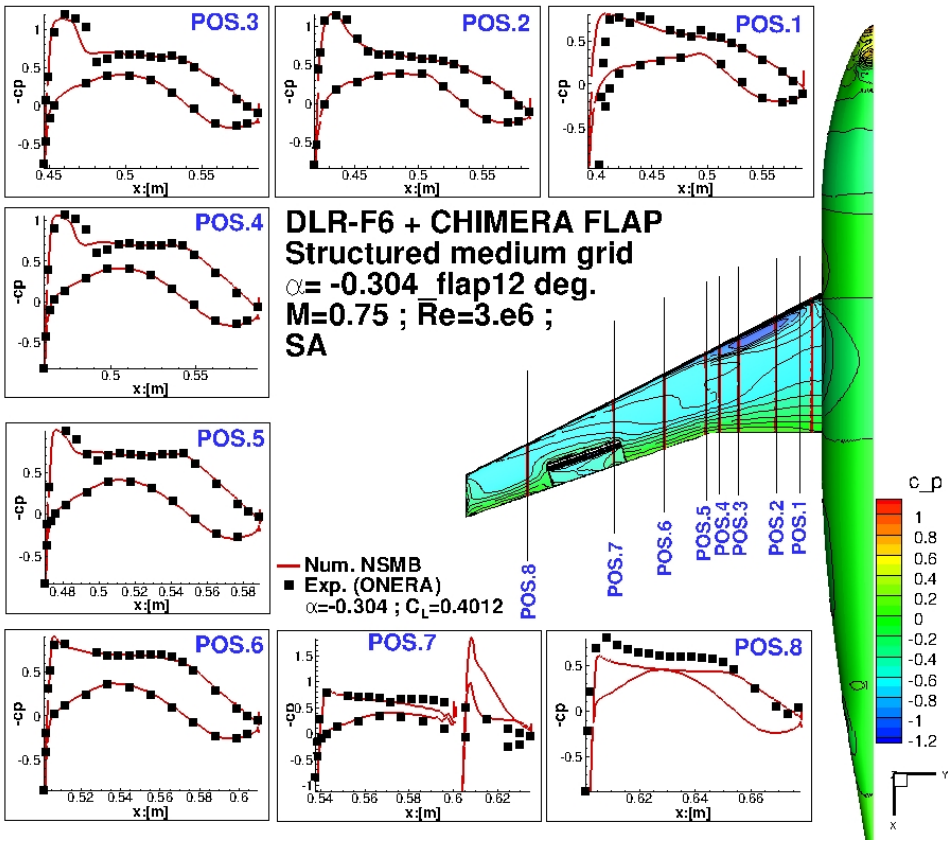


FIGURE 5.6 – Comparaison de coefficient de pression obtenu par NSMB sur maillage chimère DLR-F6 + FLAP avec données expérimentales de l'ONERA [32].

Chapitre 6

Implémentation d'un raffinement automatique par méthode chimère dans NSMB

6.1 Adaptation du Code NSMB

6.1.1 Structure des données

Historiquement la structure de stockage de données au sein du code *NSMB* est gérée à l'aide de grands tableaux globaux communs à tous les blocs du maillage. Cette approche avait l'avantage de condenser l'écriture en mémoire dans des sections continues et était donc bienvenue sur des machines où l'espace en mémoire vive pouvait être relativement restreint. Rappelons que le code *NSMB* date des années 1980.

Cependant cette méthode de stockage peut aujourd'hui poser des problèmes. En effet avec l'augmentation de la mémoire vive et l'évolution du stockage sur disque dur, les tailles des maillages s'accroissent. Malgré cette augmentation, les tailles des mémoires caches (proches des processeurs et beaucoup plus rapides) sont quant à elles beaucoup plus restreintes. De plus avec l'utilisation du parallélisme les caches de plus haut niveau sont partagés entre les différents processeurs d'une machine. L'utilisation de très grands tableaux peut donc être contre-productive, les différents processeurs luttant pour l'utilisation du cache et le temps de calcul n'est pas aussi rapide qu'escompté.

De plus cette méthode de stockage, qui peut-être adaptée pour des maillages au nombre de mailles fixes, n'est pas adaptée à l'ajout ou au retrait de nouveaux blocs de maillages comme nous souhaitons le faire. En effet garder un tel type de stockage nécessiterait à chaque ajout ou retrait d'un bloc d'allouer des espaces plus ou moins étendus en mémoire respectivement à un ajout ou retrait de bloc, puis de recopier l'entièreté des données dans ces espaces. En cas de raffinement multiples, le coût en temps de calcul d'une telle opération serait donc fortement préjudiciable à l'exécution du code.

Nous avons donc modifié la structure de données du solveur pour nous affranchir d'un

stockage continu pour tous les blocs et stocker les informations relatives aux blocs dans des tableaux dédiés comme le schématise la figure 6.1

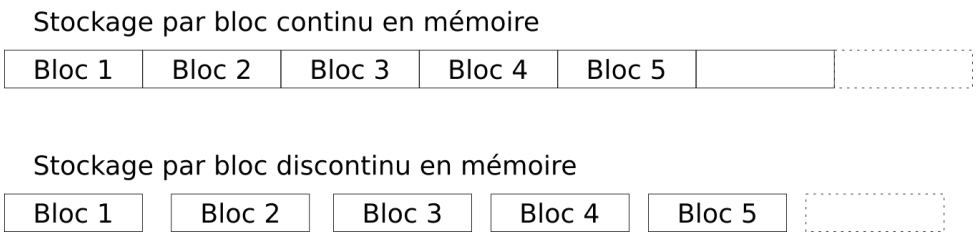


FIGURE 6.1 – Ancien stockage continu et nouveau mode de stockage discontinu par bloc introduit

Un certain nombre d'informations étant regroupé par bloc, il a donc été nécessaire de définir des types dérivés permettant d'encapsuler ces dernières. Par exemple le module définissant les variables d'état et informations sur les conditions limites se décompose comme suit dans la version originale du solveur :

```
      module statevector_addresses
!
! ... state vector, old solution vector, wall boundary arrays
!
          integer(kind=8), dimension(:,:), allocatable ::
     &                     iadw,iadwt,iadwsav,iadwold,
     &                     iadwlbc,iadinflbc
!
          real(kind=8), dimension(:), allocatable ::
     &                     w,wt,wsav,wold,wlbc,inflbc
      end module statevector_addresses
```

où les variables préfixées de *iad* sont des tableaux d'indices permettant d'identifier les positions de départ des informations relatives aux tableaux de la Table 6.1 (cf. sept pages plus loin). L'ensemble de ses tableaux n'ont plus d'utilité avec la nouvelle structure de données et sont donc supprimés. Le module est donc réécrit comme indiqué dans le listing ci-dessous.

Les encapsulations d'informations au sein de nouveaux types dérivés sont inspirées de la proposition de nouvelle architecture de données pour le code *NSMB* J. BOHBOT [64]. Cependant l'ensemble de l'arborescence a plusieurs niveaux proposée dans J. BOHBOT [64] n'est pas mise en place car nécessitant une refonte plus profonde du code.

Ces modifications préalables ont été validées sur une base de données de cas test balayant les différentes capacités de calcul du code $NSMB$.

N.B. : On prendra soin de déclarer les tableaux de pointeurs avec l'initialisation => null() afin d'éviter l'utilisation hasardeuse d'espace mémoire.

```
module statevector_addresses
  !
  TYPE PhysicStateVector
    !
    ! State vector and backups
    !
    real(kind=8), dimension(:,:,:,:), pointer :: w    => null()
    real(kind=8), dimension(:,:,:,:), pointer :: wt   => null()
    real(kind=8), dimension(:,:,:,:,:), pointer :: wold => null()
    real(kind=8), dimension(:,:,:,:), pointer :: wsav => null()
  END TYPE PhysicStateVector

  TYPE PhysicBoundaries
    !
    ! boundaries
    !
    real(kind=8), dimension(:), pointer :: wlbc   => null()
    real(kind=8), dimension(:), pointer :: inflbc => null()

  END TYPE PhysicBoundaries

  !
  type(PhysicStateVector), dimension(:,:), pointer ::
  PhyStatVec => NULL() ! contains w and wt
  type(PhysicBoundaries) , dimension(:,:), pointer ::
  PhyBCVec   => NULL() ! contains w and wt
  !
end module statevector_addresses
```

6.1.2 Allocations mémoire

Tableaux de type dérivés L'ensemble des routines d'allocation mémoire est fortement impacté par ce changement de structure de données. Elles sont réécrites en introduisant également des routines d'allocation par bloc prenant pour argument le numéro du bloc à allouer, son niveau multigrille ainsi que les paramètres ad-hoc nécessaires qui permettent la factorisation de large portion de code. Ces routines d'allocation par bloc peuvent donc être appelées deux fois, une première fois dans une boucle au sein d'une routine d'allocation globale lors de l'initialisation du calcul et une seconde fois au sein d'une routine de mise à jour permettant d'allouer la mémoire pour les nouveaux blocs créés lors du raffinement.

Tableaux indexés par bloc Les tableaux indexés par bloc, typiquement les tableaux de paramètres, sont conservés en l'état et non encapsulés dans des types dérivés comme les spécifications proposées par J. BOHBOT [64] le préconisent. La taille de ces tableaux est simplement égale au nombre de blocs du problème multiplié par un nombre de paramètres et les redimensionner n'est pas très coûteux en temps.

Redimensionnement Classiquement un redimensionnement d'un tableau T d'une taille N à $N+X$ s'opère en deux temps. Dans un premier temps une copie intégrale de l'information est stockée dans un tableau temporaire. Puis dans un second temps cette information sauvegardée est recopiée dans le tableau principal redimensionné. Ceci ce traduit en Fortran par un code du type :

```
! allocation d'un tableau temporaire
        allocate(tmp(N+X))
! memorisation des informations
        tmp(1:N) = T(1:N)
! agrandissement du tableau principal
        deallocate(T)
        allocate(T(N+X))
! recopie des donnees
    T(1:N)=tmp(1:N)
! liberation memoire temporaire
    deallocate(tmp)
```

On constate que l'information initialement contenue dans T est copiée deux fois. Cette procédure peut être améliorée en utilisant des pointeurs pour copier seulement une fois l'information de la manière suivante. Il est ainsi possible de s'affranchir de la sauvegarde intégrale du tableau initial en créant un nouveau pointeur permettant d'accéder à ce même tableau. Cet "alias" permet ainsi de libérer le pointeur principal pour allouer un nouvel espace et procéder à une seule copie d'information. Ceci ce traduit en Fortran par un code comme :

```
! memorisation des la position memoire des donnes
    tmp=>T
!  agrandissement du tableau principal
    T=>Null()
    allocate(T(N+X))
! recopie des donnees
    T(1:N)=tmp(1:N)
!  liberation memoire initiale
    deallocate(tmp)
```

6.2 Procédure de raffinement

La procédure de raffinement peut-être résumée par l'organigramme présenté fig. 6.2. Une fois les conditions de raffinement validées, un ou plusieurs processeurs se voient attribuer la création des nouveaux maillages. Une fois ces maillages créés, l'information de la simulation en cours est transférée sur ces nouveaux supports de calcul.

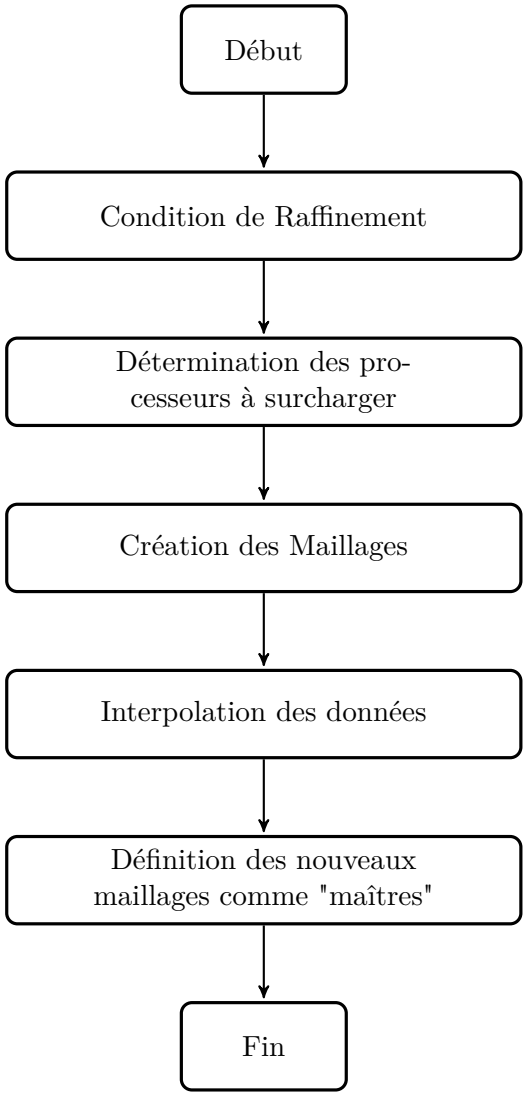


FIGURE 6.2 – Organigramme de la génération du maillage

Les différents blocs d'un maillage chimère ont chacun un niveau. Comme indiqué dans la section 5.2.2, ce niveau est le critère permettant d'imposer manuellement un bloc comme dominant ou dominé par rapport aux autres. Plus le niveau d'un bloc est grand, plus celui-ci est dominant ou "au-dessus" des autres. Au contraire plus son niveau est faible, plus il est dominé ou "en-dessous" des autres. Dans une configuration normale les blocs dont les maillages sont les plus détaillés possèdent les niveaux les plus élevés et sont les supports pour les calculs. Cependant nous introduisons lors du raffinement un ou plusieurs blocs détaillés ne possédant aucune information sur l'écoulement. Il est donc

nécessaire de ne pas définir ces blocs comme dominants. C'est pour cela que la procédure de raffinement s'opère en deux "phases". Lors de la première phase les nouveaux blocs sont insérés "en-dessous" des maillages existants de manière à pouvoir récupérer les informations de l'écoulement existant par interpolation des données existantes. La seconde phase s'effectue après un nombre de pas de temps nécessaires pour remplir les tableaux requis lors de l'intégration temporelle. Cette phase consiste simplement à placer les nouveaux blocs "au-dessus" des blocs précédents afin de les rendre dominants pour la suite du calcul.

Détaillons la procédure :

★ Phase I

- **Détermination des conditions de raffinement** : La détermination des conditions de raffinement est fixée pour le moment par l'utilisateur au sein du fichier d'entrée.
- **Choix du processeur à surcharger** : le processeur étant le moins sollicité est par défaut celui accueillant le nouveau bloc créé. Dans le cas de la création simultanée de plusieurs blocs cette procédure doit être optimisée pour répartir efficacement la charge de travail.
- **Mise à jour du nombre et numéros de blocs**
- **Initialisation de la nouvelle base de donnée** : par copie de l'existant
- **Mises à jour de la structure de données** : la mise à jour de la structure de données nécessite une reprise partielle des fonctions d'initialisation exécutées au lancement du solveur.
- **Interpolation des nouveaux blocs** : l'initialisation des nouveaux blocs créés est effectuée en relevant l'ensemble des niveaux chimère de tous les blocs existants de manière à placer le nouveau bloc "sous" ceux existants et ainsi effectuer l'interpolation des champs existants vers ce dernier.
- **Mise à jour des données chimères et ibm**
- **Mise à jour des champs annexes**
- **Écriture des données dans une nouvelle base de données**
- **Verrouillage du processus de raffinement** : Le bloc nouvellement créé étant encore "en-dessous" des précédents, de nouveaux raffinements sont donc suspendus le temps de finaliser le processus.

★ Phase II : cette passe est effectuée après un nombre d'itérations nécessaires permettant l'intégration temporelle d'être effective sur les nouveaux blocs.

- **Relèvement du niveau chimère du bloc raffiné**
- **Mise à jour des données chimères et ibm**

- **Dé-verrouillage du processus de raffinement** : un nouveau raffinement peut être maintenant opéré

6.2.1 Conditions de raffinement

6.2.2 Types de raffinement

Maillages Prédéfinis Une première méthode pour raffiner une paroi ou une zone est de définir une boite englobant l'espace à raffiner. Un maillage cartésien peut y être facilement défini à partir des paramètres utilisateurs ou en estimant localement la taille des mailles à prendre en compte. Cette approche est relativement aisée à mettre en place en 2 et 3 dimensions. Des exemples de génération de maillages sont illustrés sur la figure 6.3

La définition de tel maillages appliqué au raffinement d'une paroi peut introduire un grand nombre de mailles "inutiles" et de multiples superpositions augmentant grandement la taille du maillage.

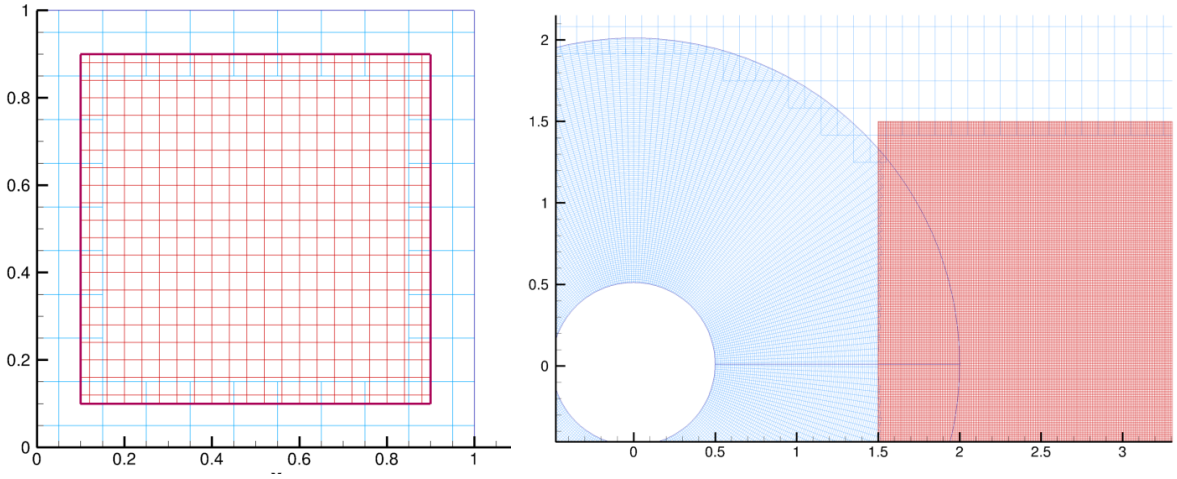


FIGURE 6.3 – Exemples de raffinements définis[2] par l'utilisateur. En rouge les nouveaux blocs

Maillages de paroi Une seconde méthode introduite plus spécifiquement pour raffiner des parois est de garder l'esprit du raffinement par bloc et d'y adjoindre la prise en compte locale de la direction normale globale d'une portion de celle-ci.

Maillages Conformes à la paroi Une dernière méthode pour raffiner plus finement une paroi est de prendre en compte pour chaque segment de paroi la normale à celle-ci. Il est ainsi possible de réduire grandement le nombre de superpositions des nouveaux maillages créés voire de totalement les supprimer à l'aide de communications de blocs. Des traitements spéciaux doivent cependant être pris en compte lorsque la géométrie possède

des arêtes comme par exemples celles classiques d'ailes d'avion. Des exemples de cette méthode sont illustrés par le raffinement du maillage IBM d'un cylindre (cf. fig. 6.4) ainsi que celui d'une aile d'avion (cf. fig.6.5).

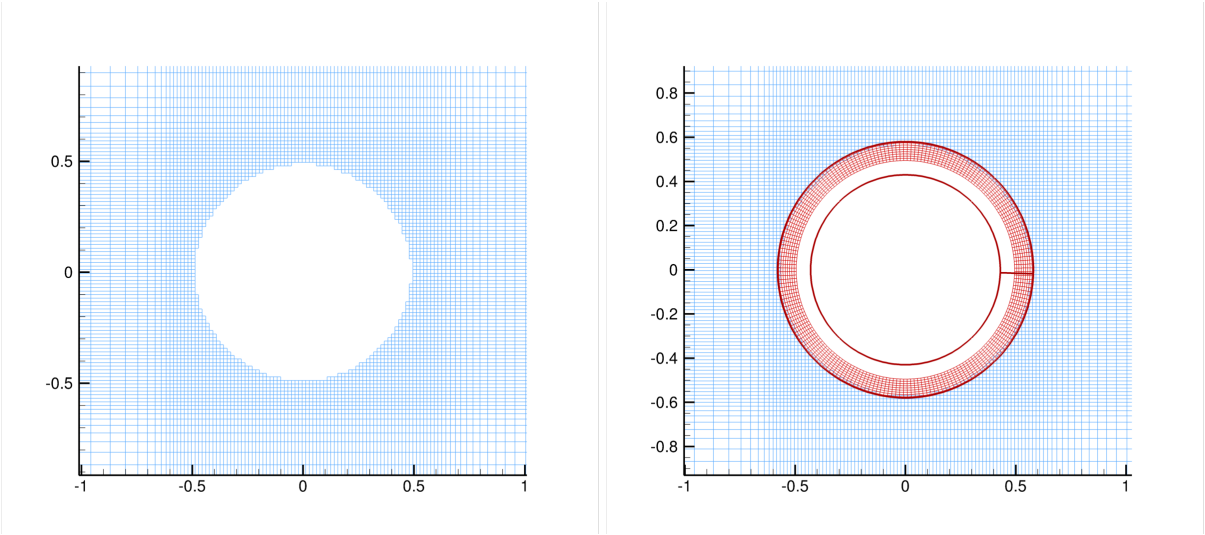


FIGURE 6.4 – Raffinement d'un maillage IBM de cylindre.

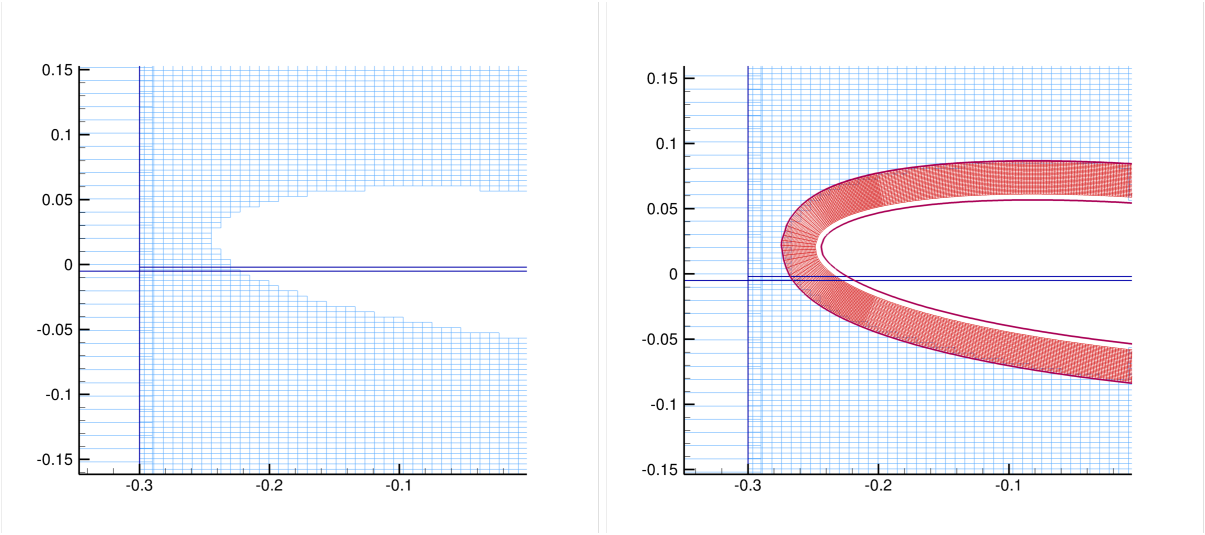


FIGURE 6.5 – Raffinement d'un maillage Chimère/IBM d'aile.

Maillages externes Enfin l'utilisateur peut également prédéfinir totalement une portion de maillage plus détaillée stockée dans une base de données séparée de celle de départ. Ce maillage sera alors lu lors de la procédure de raffinement et ainsi intégré au calcul.

6.2.3 Paramètres de raffinement

Dans cette section nous détaillons les paramètres utilisateurs disponibles pour générer les maillages raffinés. Les principaux paramètres déterminent le type de génération de

maillage, les paramètres déclenchant le raffinement ainsi que le mode de surcharge des processeurs. Les indications entre parenthèses précisent le type du paramètre : **(i)** pour entier **(f)** pour réel, **(l)** pour un booléen ; préfixé d'un nombre indiquant la taille si un tableau de valeurs est nécessaire.

Paramètre et Valeurs	Description
fchim Refinement Generation (i)	Sélection du mode de création du maillage
input	lecture des paramètre de raffinement dans le fichier d'input
memcom	lecture d'un bloc préalablement sauvegardé dans une base memcom
auto	raffinement totalement automatisé
fchim Refinement Load Balance (i)	Sélection du mode de surcharge processeur
auto	le processeur ayant la moindre charge de travail traitera le nouveau bloc
proc	le processeur déterminé par la variable ipm current autorefine est surchargé
fchim Refinement Condition (i)	Sélection de la condition de raffinement
auto	automatique
step fixed	pas de temps fixé
step repeat	pas de temps fixé + répétition
time fixed	temps donné
time repeat	temps donné + répétition

TABLE 6.1 – Paramètres principaux du raffinement

Raffinement Utilisateur

Lorsque **(fchim Refinement Generation = input ou memcom)** l'utilisateur fixe certains paramètres du raffinement en avance.

Paramètres temporels : l'utilisateur peut choisir de fixer le pas de temps ou l'instant auquel le raffinement est déclenché ainsi que des intervalles de répétition avec les paramètres suivants.

Paramètre et Valeurs	Description
chimrc nstep (i)	itération du premier raffinement
chim dnstep (i)	intervalle d'itérations entre deux raffinements
chimrc time (f)	temps du premier raffinement
chimrc dtime (f)	intervalle de temps entre deux raffinements

TABLE 6.2 – Paramètres temporels

Paramètres de grille : les paramètres décrits dans la Table 6.3 permettent de générer les nouveaux maillages.

Mouvement de maillage : les paramètres ALE décrits dans la Table 6.4 fixent les valeurs de déplacement des nouveaux maillages

Paramètre et Valeurs	Description
fchim refinement type (i)	Selection du mode prédeterminé de maillage
squared	
cylinder	
chimRef ni (i)	nombre de cellules à créer dans la direction i
chimRef dni (i)	taille des cellules dans la direction i
chim Ref [XYZ]0 (f)	position de référence du bloc
fchimRepeatOffset (i)	flag d'activation de la répétition spatiale de la création de nouveau maillage
chimRef d[XYZ]repeat (f)	espacement les positions de référence des nouveaux blocs créés

TABLE 6.3 – Paramètres de grille et temporels

Paramètre et Valeurs	Description
chmRef ALE (i)	type ALE pour les nouveaux blocs créés
chimRef ALE Velo (3f)	vitesse imposée des blocs raffinés
chim Ref ALE AngVelo (3f)	vitesse angulaire imposées sur les blocs raffinés

TABLE 6.4 – Paramètres liés au mouvement du maillage

6.3 Communications

À chaque itération, des échanges d'informations liées à la gestion de maillage chimère sont opérés. Ces échanges sont opérés simplement dans NSMB via la librairie MPI (Message Passing Interface) en rassemblant l'ensemble des informations calculées par chaque processus "esclave" au niveau du processus "maître" puis en diffusant l'information à l'ensemble des processus. Ces communications sont illustrées dans la figure 6.6. On voit ainsi qu'un grand nombre d'informations inutiles - en blanc dans la 6.6 - peuvent être échangées. Dans notre exemple, il est à noter que les processus 2 et 4 ne gérant pas de bloc possédant une superposition chimère communiquent quand même un tableau vide. D'autre part, les processus 0,1 et 3 gérants des blocs superposés, communiquent également une grande partie d'information inutile.

Dans cette partie nous détaillerons plusieurs approches envisagées pour diminuer les échanges des informations.

6.3.1 Communications globales chimère

Une première approche consiste simplement à réduire le volume d'informations communiquées en restreignant celles-ci uniquement au processus maître ainsi qu'aux processus gérant un bloc chimère ou possédant une superposition avec un bloc chimère. Dans notre exemple illustré par la figure 6.7, on supprime simplement les communications provenant

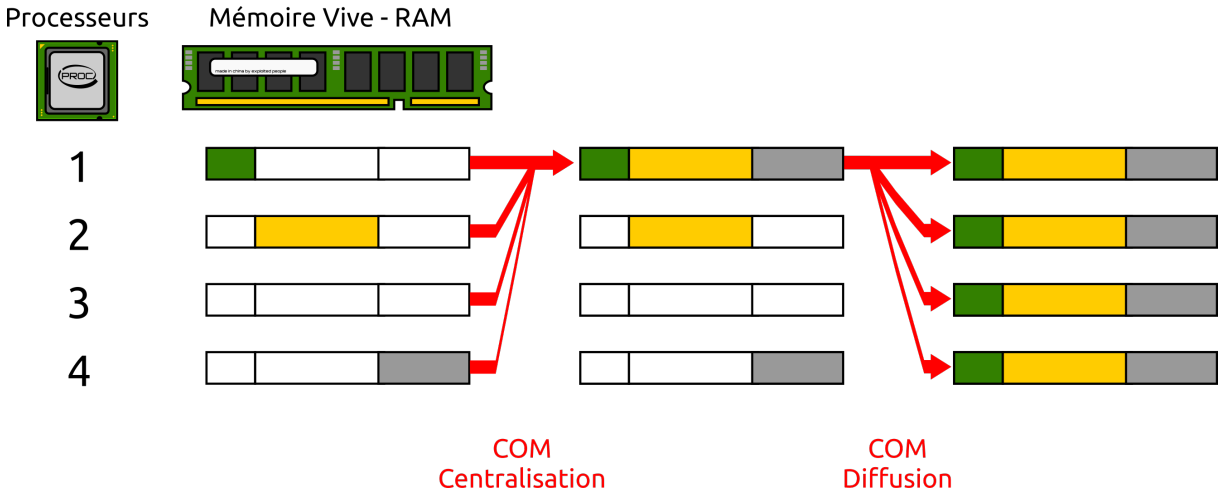


FIGURE 6.6 – Schéma de la communication de base des informations chimères

et en direction des processus 2 et 4.

Dans un premier temps il est donc nécessaire de déterminer quels sont les processus qui doivent communiquer. Dans le code *NSMB* cette information est extraite du tableau *ncccpblk* qui permet de savoir quels blocs se superposent. Cette information extraite permet de définir un espace de communication restreint. Dans notre exemple l'espace de communication global défini classiquement par *MPI_COMM_WORLD* et rassemblant tous des processus est réduit à un espace *newcomm* contenant uniquement les processus 0,1 et 3. Ce nouvel espace de communication est obtenu à l'aide de la fonction *MPI MPI_Comm_Split*. On note également que dans le cas où les blocs de maillages sont mobiles, l'espace de communication restreint peut être amené à changer. L'algorithme de communication devient alors :

for *liste des bloc* **do**
  Initialisation de la couleur du bloc basé sur les informations de superposition de bloc
end
if *Couleurs différentes de la précédente communication* **then**
  Mises à jour de l'environnement de communication
  newcomm ← MPI_Comm_Split
end
if *Besoin de communiquer* **then**
  Échange d'informations uniquement entre les processus colorés
  MPI_ALL_REDUCE(newcomm)
end

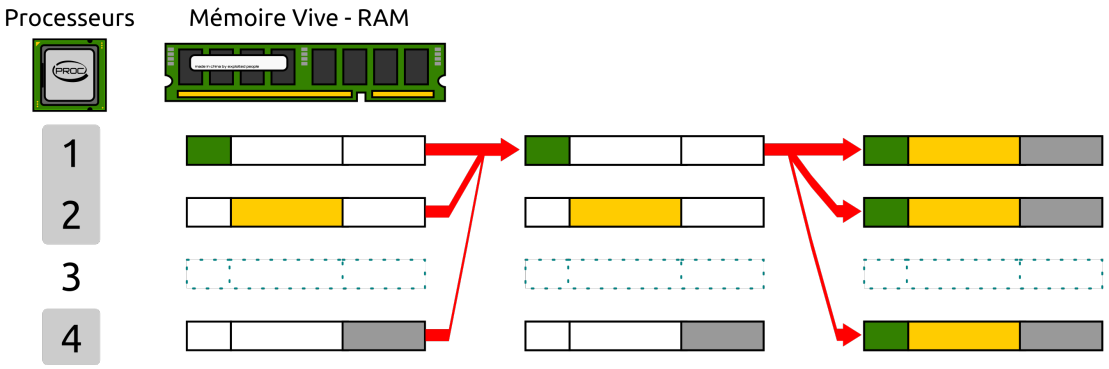


FIGURE 6.7 – Schéma de la communication de base optimisée simplement

6.3.2 Communications globales chimère restreintes

La proposition de la première méthode a motivé l'implémentation d'une nouvelle amélioration par J.B. Vos de CFD Engineering au sein de la version de NSMB utilisant le stockage par tableaux "globaux". Le principe reste dans l'idée identique à la première optimisation en y ajoutant une communication gérée plus finement pour ne communiquer au processeur maître que la partie des informations nécessaires. Cette communication est illustrée par la figure 6.8.

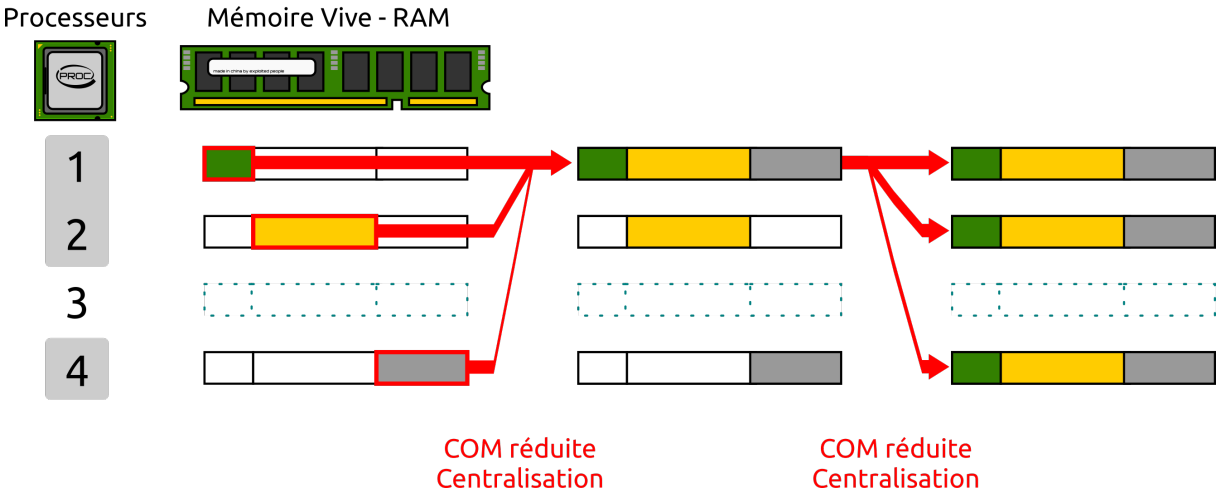


FIGURE 6.8 – Schéma de la communication de base optimisée plus finement

6.3.3 Perspectives

Dans les précédentes méthodes chaque processus possédant une information doit procéder à une ou plusieurs communications car les informations sont dupliquées sur chaque processus. Cependant chaque processus ne met à jour que les informations relatives aux blocs qu'il gère. Il est donc envisageable sur une même machine de partager l'espace mémoire au lieu de dupliquer l'information et de réaliser non pas des échanges entre processus

mais entre nœud de calcul comme c'est schématisé dans la figure 6.9. Les communications peuvent donc ainsi être divisées au maximum par le nombre de processus présents sur un nœud de calcul. Pour le cas où la simulation est lancée sur une unique machine, le besoin de communication de données est réduit à une simple synchronisation et l'espace mémoire est également économisé.

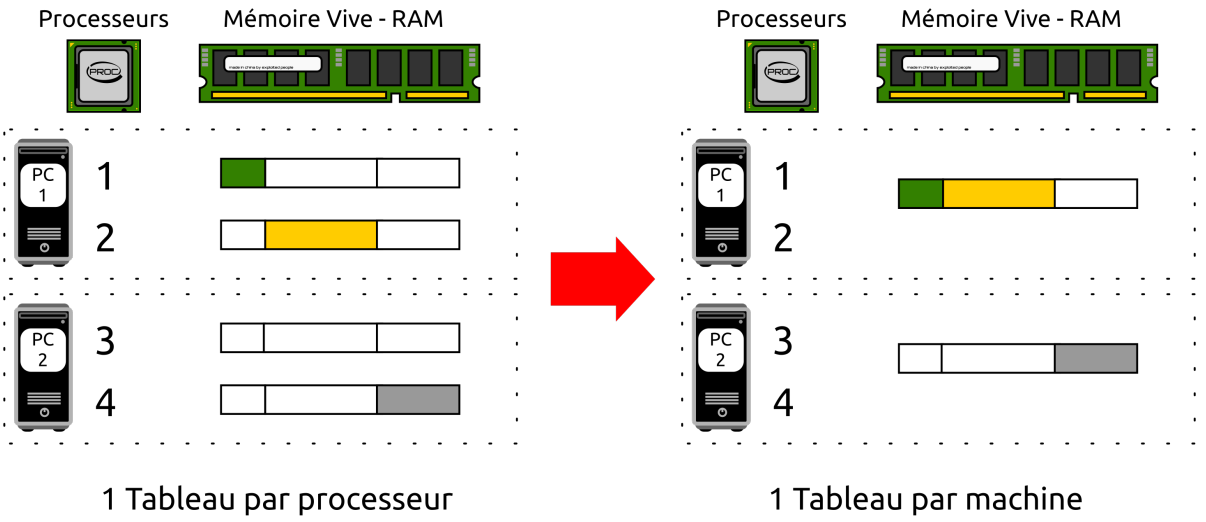


FIGURE 6.9 – Proposition de communication chimère avec prise en compte de la localité

6.4 Parallélisme

6.4.1 Répartition de charge

L'introduction de nouveaux maillages raffinés lors du calcul requiert de répartir la charge de travail entre les différents processeurs. Actuellement deux cas sont pris en compte. Le premier consiste à surcharger un processeur unique défini par l'utilisateur et ceci pour répondre aux besoins de développement. Le second consiste à évaluer le temps de calcul d'un pas de temps consacré par chaque processeur. Le processeur ayant la charge de travail la moins importante est celui qui sera chargé de la simulation de l'ensemble des nouveaux blocs créés lors de ce pas de temps. Si de nouveaux blocs doivent être créés plus tard, un autre processeur pourra être surchargé.

6.4.2 Conclusion

Les performances des différentes approches envisagées précédemment dépendent du type de situation prise en compte, de la répartition de charge entre les processus et du nombre de communications à effectuer. L'idéal serait donc de pouvoir mesurer les performances de chaque type de communication pour chaque cas afin de pouvoir déterminer

dynamiquement la meilleure méthode à utiliser pour finir le calcul. De plus, les communications détaillées précédemment font intervenir uniquement des processus MPI et ne prennent pas en compte la localité des mémoires. En effet plusieurs processus peuvent opérer sur une même machine physique. Il est donc envisageable de mettre à profit cette localité en utilisant des tableaux partagés en mémoire RAM et ceci afin de diminuer d'autant plus les communications MPI ainsi que les besoins en mémoire dynamique.

Chapitre 7

Autres améliorations du NSMB

Au cours es travaux, il s'est avéré utile de faire évoluer le code sur plusieurs autres points.

7.1 Compilation

La compilation du code *NSMB* était jusqu'à présent gérée manuellement à l'aide d'une série de scripts de compilation dont le principal était le script *doinstall* couplé avec un fichier de configuration *doinstall.in*, le deux placés dans le répertoire *Install*. La création des fichiers objets se faisait dans un répertoire nommé *NSMB* présent au même niveau que les répertoires contenant les sources du code.

Cette méthode de compilation pose un premier souci de maintenance de diverses versions du code. En effet, il arrive que, pour des besoins de tests, plusieurs versions du code source de NSMB soient dupliquées. À cette duplication s'ajoutent les versions mineures et majeures du code délivrées régulièrement. Il en résulte une présence de nombreux fichiers sources parfaitement identiques ou différents uniquement par de simples indentations ou commentaires sur la machine.

Un second problème de cette méthode de compilation est la mauvaise captation de l'ensemble des dépendances du code ce qui fait que le processus de compilation n'arrive parfois pas à son terme et doit être relancé à de multiples reprises.

À sa décharge cette méthode de compilation permet assez simplement de recompiler un seul et unique fichier, ou quelques fichiers choisis avec des paramètres de compilation différents, notamment pour les forcer en mode de débogage.

Pour résoudre les problèmes évoqués j'ai choisi de réécrire le script de compilation en me basant sur l'outil *CMake* qui permet

- de gérer correctement la résolution de l'arbre de dépendances des fichiers sources,
- de compiler des versions du même code avec des options de compilations différentes sans duplication du code source d'origine.

À ces résolutions de problèmes s'ajoute de nouvelles fonctionnalités permettant :

- d'informer l'utilisateur avant le début de la compilation de problèmes de dépendances ou de chemins non existants,
- de gérer des dossiers de codes sources restreints contenant des modifications. Nous développerons plus avant cette fonction par la suite,
- de compiler un unique utilitaire d'*NSMB* ou tous les utilitaires.

7.1.1 Fichier de configuration

Comme pour la version de compilation précédente, un certain nombre de paramètres doivent être fournis lors du processus de compilation. Un fichier de configuration d'exemple est placé à la racine du dossier du code *NSMB* nommé *custom_parameters.cmake*. Ce fichier reprend évidemment les définitions présentes dans *doinstall.in*.

Ce fichier *custom_parameters.cmake* présent à la racine sera à copier dans le dossier de compilation comme nous le verrons dans la section suivante !

```
SET(CUSTOM_PROJECT_DIR   "$ENV{HOME}/Projet-NSMB")
```

CUSTOM_PROJECT_DIR définit le répertoire contenant le dossier des sources de *NSMB*. Les librairies peuvent également être placées dans ce dossier. La variable $\$ENV\{HOME\}$ pointant sur le dossier utilisateur, permet de s'affranchir de l'adaptation de ce chemin si l'arborescence est conservée d'une machine à une autre ou d'un utilisateur à un autre.

```
SET(FC  "ifort")
```

FC définit le compilateur à utiliser. La variable *FC* peut prendre les valeur ifort, gfortran ainsi que openf95 et pgf dont les options de compilations ont seulement été reportées depuis le *doinstall.in*

```
SET(CMAKE_BUILD_TYPE DEBUG)
```

CMAKE_BUILD_TYPE définit le mode de compilation à adopter. Les valeurs suivantes définies :

- **RELEASE** : pour obtenir un exécutable optimisé pour la vitesse d'exécution en production.
- **DEBUG** : pour déboguer le code avec le maximum d'options de débogage. L'exécutable généré est donc extrêmement lent.

- **RELWITHDEBINFO** : pour obtenir un exécutable permettant un équilibre entre temps d'exécutions et informations de débogage.

Les options de compilations sont définies directement dans le fichier *CmakeLists.txt*. L'utilisateur avancé pourra y modifier les valeurs par défaut.

```
SET(CUSTOM_PREFIX_BUILD_DIR  "")
```

CUSTOM_PREFIX_BUILD_DIR permet d'adapter le dossier de sortie des exécutables créés. Par défaut le dossier de sortie de l'exécutable NSMB est construit comme suit :

CUSTOM_PROJECT_DIR/
NSMB-EXE-<NUMERO-VERSION>/
<CUSTOM_PROJECT_DIR>/
{ bin,lib } /
<CUSTOM_PREFIX_BUILD_DIR>/
<ARCH>/
<CMAKE_BUILD_TYPE>

avec le numéro de version qui est défini dans le fichier *CMakeLists.txt*. Le dernier sous-répertoire référençant l'option de compilation est omis pour la compilation en mode production.

De multiples paramètres sont ensuite disponibles et activables en fonctions des besoins des utilisateurs

- IJK : activé par défaut. L'accès aux valeurs de champs se fait par indice i,j,k. Si non activé la version vecteur de NSMB est alors compilée.
- SWAPPED : permute l'indice d'accès,
- FSI : active la gestion d'intéraction fluide structure,
- MPI : active la version parallèle avec Message Passing Interface,
- MPP : active la librairie Mutation++,
- UTIL : active la possibilité de compiler les utilitaires,
- MEMCOM_VERSION : la version 7.0 est activée par défaut. L'utilitaire de découpage de maillage en bloc MBsplit (voir plus loin) ayant été porté sur la version 7 de memcom les versions 6.X de memcom (stockage en fichier) peuvent donc être transformées en version 7.X (stockage en dossier). L'option est gardée pour la compatibilité et les valeurs 6.5 et 6.6 sont acceptables si la librairie MEMCOM est présente.
- DOC : active la possibilité de générer la documentation de NSMB,
- DOTS : permet de générer les graphes de fonctions appelées et fonctions d'appel dans la documentation. La librairie *graphviz* doit alors être installée.

Dossiers de modifications

Une nouvelle fonctionnalité importante a également été introduite. La possibilité de spécifier un dossier de modifications. Avec la méthode de compilation cmake nous évitons déjà de dupliquer ou de recompiler notre code pour obtenir un exécutable de production ou un exécutable de débogage. Cependant quand l'utilisateur souhaite tester des modifications, le code doit encore être dupliqué. Pour pallier a ce dernier problème nous avons introduit la possibilité de spécifier un dossier reprenant l'arborescence - qui peut être partielle - du code *NSMB*, dans lequel sont contenus uniquement les fichiers sur lesquels l'utilisateur effectue des modifications. Si des fichiers aux noms identiques sont présents dans le dossier principal, les fichiers contenus le dossier de modifications seront ceux pris en compte lors de la compilation. L'option dans le fichier *custom_parameters.cmake* est la suivante :

```
SET(CUSTOM_MODIFS_DIR    "/chemin/vers/le/dossier/de/modifications" )
```

Pour aller plus loin, ces modifications peuvent être compilées différemment du code original, notamment pour obtenir les informations de débogage.

```
SET(CUSTOM_MODIFS_DIR_BUILD_TYPE DEBUG)
```

7.1.2 Utilisation

Une fois le fichier de configuration correctement paramétré en fonction des souhaits de l'utilisateur.

1. Créer un dossier de compilation, par exemple **BUILD**, placé dans le répertoire de votre choix.
2. Copier le fichier de configuration paramétré *custom_parameters.cmake* dans ce dossier créé.
3. Créer le *makefile* en adaptant les chemins et en exécutant la commande

   ```
   cmake <CUSTOM_PROJECT_DIR>/nsmb<VERSION>
   ```

4. Compiler en tenant compte du nombre de processeurs présents sur votre machine - typiquement $n + 1$:

   ```
   make -j 9
   ```

7.2 Code

7.2.1 Conversion au format libre

Le code NSMB était jusqu'à présent écrit au format fixe Fortran et donc limité à 72 caractères par lignes. Un utilitaire de conversion a été mis en place permettant de convertir NSMB au format libre tout en permettant quelques nettoyages. Cet utilitaire est basé sur l'utilitaire de conversion proposé par de James Goppert et Lenna Peterson [45], utilitaire qui a été amélioré et adapté pour une application au code *NSMB*. Des scripts additionnels permettent

- d'égaliser les normes d'écritures notamment concernant les notations **.EQ.** et == et autres équivalents (*clean-nsmb.sh*), à l'aide d'expressions régulières.
- de terminer correctement les fin de fonctions (*end.sh*) pour les fichiers contenant une unique fonction,
- de forcer l'indentation du code (*emacsfix.sh*) qui ne l'est pas du fait de la limitation du format fixe,
- d'intervertir ou de mieux terminer des fins de boucles do emboitées dans des conditions préprocesseurs et mal gérés par les éditeurs de texte.

Certains fichiers restent cependant non traités par le script du fait de la présence intensive de code "spaghetti indigeste". L'outil de conversion donne finalement quelques statistiques sur le nombre d'instruction considérées comme remplaçables comme les **GOTO** et autre *IF arithmétiques*.

7.3 Scripts

Plusieurs utilitaires ont été développés ou améliorés au cours de nos travaux.

7.3.1 M2T

L'utilitaire *m2t* pour *memcom to tecplot* est l'utilitaire quotidien d'extraction des données stockées au sein des bases de données memcom vers un fichier lisible par le visualiseur tecplot.

Gain d'espace disque Cet utilitaire permettait de créer des fichiers texte lisibles par tecplot. Hors à l'ouverture de ce logiciel de visualisation, le fichier de données est automatiquement converti vers un fichier temporaire au format binaire plus compact. La

différence d'espace de stockage nécessaire entre un fichier texte tecplot et un fichier binaire tecplot est d'un facteur 6.

Afin de gagner du temps d'exécution en économisant les conversions multiples et de l'espace disque, l'utilitaire tecplot a été augmenté de la fonction d'écriture au format binaire tecplot directement. La compilation de l'utilitaire requiert donc la présence de la librairie tecplot adhoc permettant de réaliser les appels nécessaires.

User Friendlyness Une deuxième amélioration a également été ajoutée via la création d'un lanceur python *m2t.py*. En effet, l'utilitaire *m2t* est écrit en Fortran et ne permet pas des interactions utilisateurs "user friendly" en ligne de commande. Par défaut *m2t* pose de nombreuses questions pour lesquelles la réponse par défaut est le plus souvent choisie. Les seuls paramètres pertinents restant pour une utilisation quotidienne, le nom de la base de donnée ainsi que les numéros de grille et de cycle sur lesquels les solutions sont stockées. Par exemple pour extraire les données memcom contenues dans le cycle 42 de la grille 11 dans la base *mabasededonnees.db.mc* l'utilisateur lancera la commande :

```
m2t.py mabasededonnees.db.mc -g 11 -c 42
```

où le nom de la base de données pourra être autocomplété en ligne de commande. De plus pour des cas basiques où la grille est $g = 0$ l'utilisateur pourra omettre de spécifier le numéro de grille. La spécification du cycle lorsque qu'il est égal à $cg + 1$ peut également être omis.

L'ensemble des options peut être facilement obtenue en lançant classiquement l'aide comme suit :

```
m2t.py --help
```

Allégements pour la visualisation Les domaines de calculs peuvent être très grands comparés aux zones principales d'intérêts dans une simulation. À cet effet nous avons intégré une fonctionnalité permettant de réduire le domaine de visualisation et ainsi gagner d'autant plus de place et de vitesse de création. Cette fonctionnalité trouve spécialement sont intérêt comme nous l'avons évoqué avec la gestion de domaines de calculs très étendus ainsi que pour la création de vidéos sur des domaines de calculs moyens à très étendus. Cette fonctionnalité repose sur deux approches. La première de ces approches consiste à simplement omettre des blocs dans la sauvegarde du fichier de visualisation. Par exemple l'utilisateur pourra exécuter la commande suivante pour extraire uniquement les blocs 11 et 38 du cycle 1, grille 0 de *mabasededonnees.db.mc* :

```
m2t.py mabasededonnees.db.mc -b 11 38
```

Cependant si la liste de blocs est longue, celle-ci pourra être avantageusement stockée dans un fichier

```
m2t.py mabasededonnees.db.mc --fblcks monFichierDeBlocsASauver
```

La seconde approche consiste à indiquer un fichier spécifiant des plans d'exclusion

```
m2t.py mabasededonnees.db.mc --fblcks monFichierDePlansDExclusion
```

7.3.2 Vidéos

Afin de pouvoir réaliser des vidéos, les scripts *autoscript_m2t* et *autoscript_film* ont été développés conjointement avec Dorian Pena. Ils permettent respectivement d'automatiser la conversion vers les fichiers de visualisation tecplot et d'automatiser la création d'une ou de plusieurs séquences vidéos.

Pour la partie *autoscript_film* l'utilisateur doit tout d'abord définir les paramètres de visualisation nécessaires à la création des images des vidéos par la création d'un script *mcr* tecplot. Plusieurs images de champs différents peuvent être définies par l'utilisateur au sein de ce script tecplot. Une fois l'utilisateur satisfait des options de visualisation *autoscript_film* peut être exécuté en indiquant dans l'entête du script ou directement en ligne de commande le script définissant les paramètres de visualisation.

Pour améliorer le temps de rendu il est possible d'exécuter de manière parallèle le script notamment sur la partie de génération des images. Pour ce faire la première partie du script consiste à créer autant de copies nécessaires du script de visualisation en modifiant les noms de fichiers de sortie pour chaque frame. Plusieurs instances de tecplot peuvent alors être lancées simultanément pour produire le rendu des images.

7.3.3 MBSPLIT

Le script de découpage en blocs MBSPLIT a été porté vers la version 7 de memcom. Cette conversion permet de rendre obsolète l'utilisation de la librairie memcom 6 qui possède une limite quant au stockage mémoire des bases de données.

Troisième partie

Résultats

Chapitre 8

Cavité entraînée 2D

8.1 Description

La cavité entraînée (*Lid-driven cavity*) est un cas test simple permettant de valider la bonne implémentation d'un solveur de Navier-Stokes en s'appuyant sur les l'article [48]. La configuration plane est composée d'un simple carré ide côté L dont toutes les parois sont fixes exceptée la parois supérieure pou laquelle la vitesse est fixée constante égale à U. Le mouvement de la paroi supérieure permet la génération d'une recirculation principale du fluide au centre de la cavité accompagnée de recirculations secondaires dans les angles dépendants du nombre de Reynolds.

La séparation et le ré-attachement de fluides turbulents sont des phénomènes communs en ingénierie. Ces phénomènes sont observables dans des systèmes de diffusion, de combustion ou plus généralement sur dans les écoulements autours d'ailes d'avions. Avec l'augmentation du nombre de Reynolds, il est classiquement connu que ces phénomènes sont accentués. Le nombre de Reynolds est ici basé sur la vitesse d'entraînement de la paroi supérieure :

$$\mathrm{Re} \;=\; \frac{UL}{\nu}. \tag{8.1}$$

où ν est la viscosité cinématique du fluide. Un schéma numérique implicite d'ordre deux a été utilisé pour cette simulation. La figure 8.1 illustre ce cas.

De part sa géométrie très simple, le maillage et les conditions initiales sont aisément mis en place. On notera de plus l'avantage qu'aucune spécificité chimère initiale n'est nécessaire. Cette particularité nous permettra notamment de valider l'activation du raffinement chimère en cours de simulation. Malgré cette simplicité ce cas test permet d'apprécier la non-linéarité de l'équation de Navier-Stokes.

Le maillage initial représenté sur la figure 8.2 est composé de 28x28 cellules au total. les bords sont maillés par une rangée de 10x8 cellules rectangulaires de 0.01x0.1, la partie

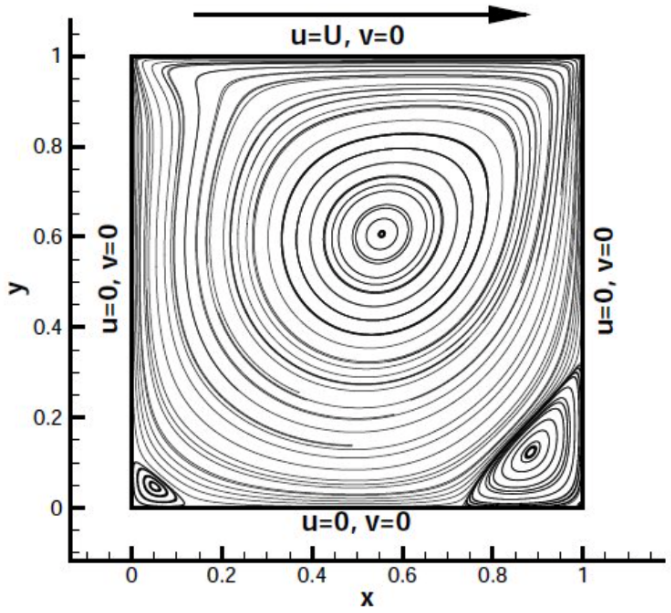


FIGURE 8.1 – Configuration de la cavité entraînée en 2D

centrale par 8x8 cellules carrées de 0.1x0.1, les coins par 10x10 cellules carrées de 0.01x0.01.

Le bloc de maillage raffiné représenté en rouge sur la figure 8.3 est composé d'une grille uniforme de 46x46 de mailles carrées de dimension 0.2 espacé de 0.04 des bords du domaine.

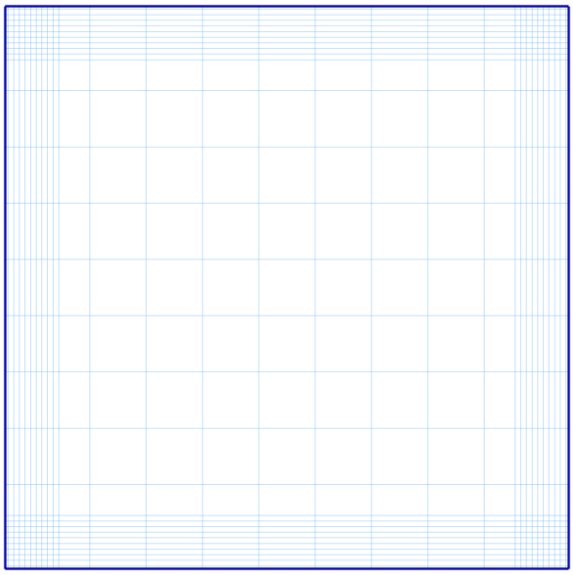


FIGURE 8.2 – Maillage initial

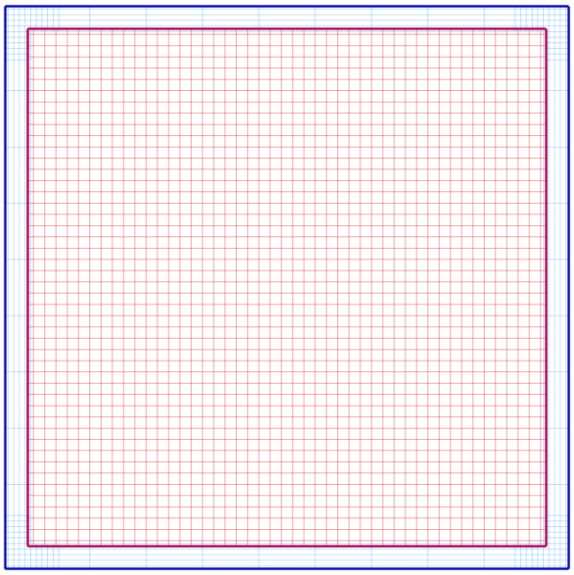


FIGURE 8.3 – Maillage raffiné

8.2 Résultats

L'article [48] fournit une description autant qualitative que quantitative des résultats de la cavité entraînée, notamment les données dimensionnelles des diverses recirculations. La figure 8.4 est extraite de l'article et illustre ce propos. Les profils de vitesses dans les plans médians calculés par le code NSMB sont comparés avec l'article [48] dans la figure 8.5.

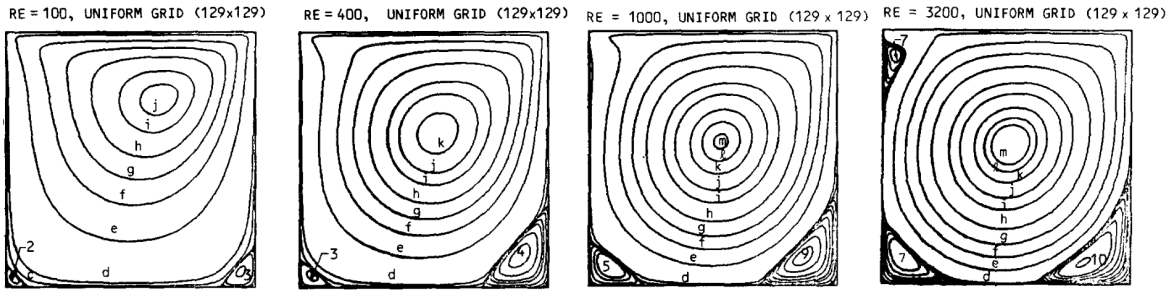


FIGURE 8.4 – Cavité entrainée : données extraites de [48] pour les Reynolds 100, 400, 1000, 3200 sur une grille uniforme 129x129.

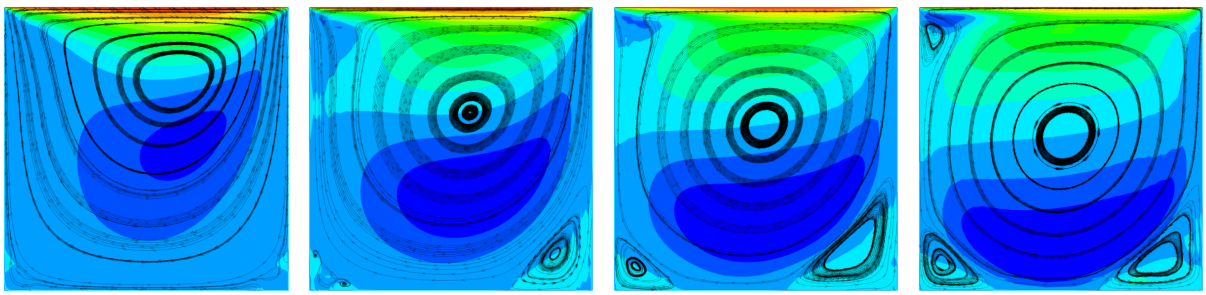


FIGURE 8.5 – Cavité entrainée : champs de vitesse horizontale u obtenus sur le maillage raffiné 8.3

Les profils de vitesses sur les axes médians $x = 0.5$ et $y = 0.5$ sont comparés avec les résultats présentés dans [48]. Les figures 8.6 présentent les résultats sur le maillage grossier qui permettra d'initialiser le calcul du maillage raffiné. Les résultats sur le maillage grossier sont évidemment très grossièrement en accord avec ceux de [48], mais permettent d'obtenir une initialisation rapide du champ de vitesse. Les résultats obtenus par la suite sur le maillage raffiné sont en accord avec ceux de [**ghia182**] et concordent avec ceux obtenus sur le maillage témoin initialement maillé finement.

Cette étude permet la validation du code dans le cas suivant : écoulement bidimensionnel avec raffinement calculé en séquentiel.

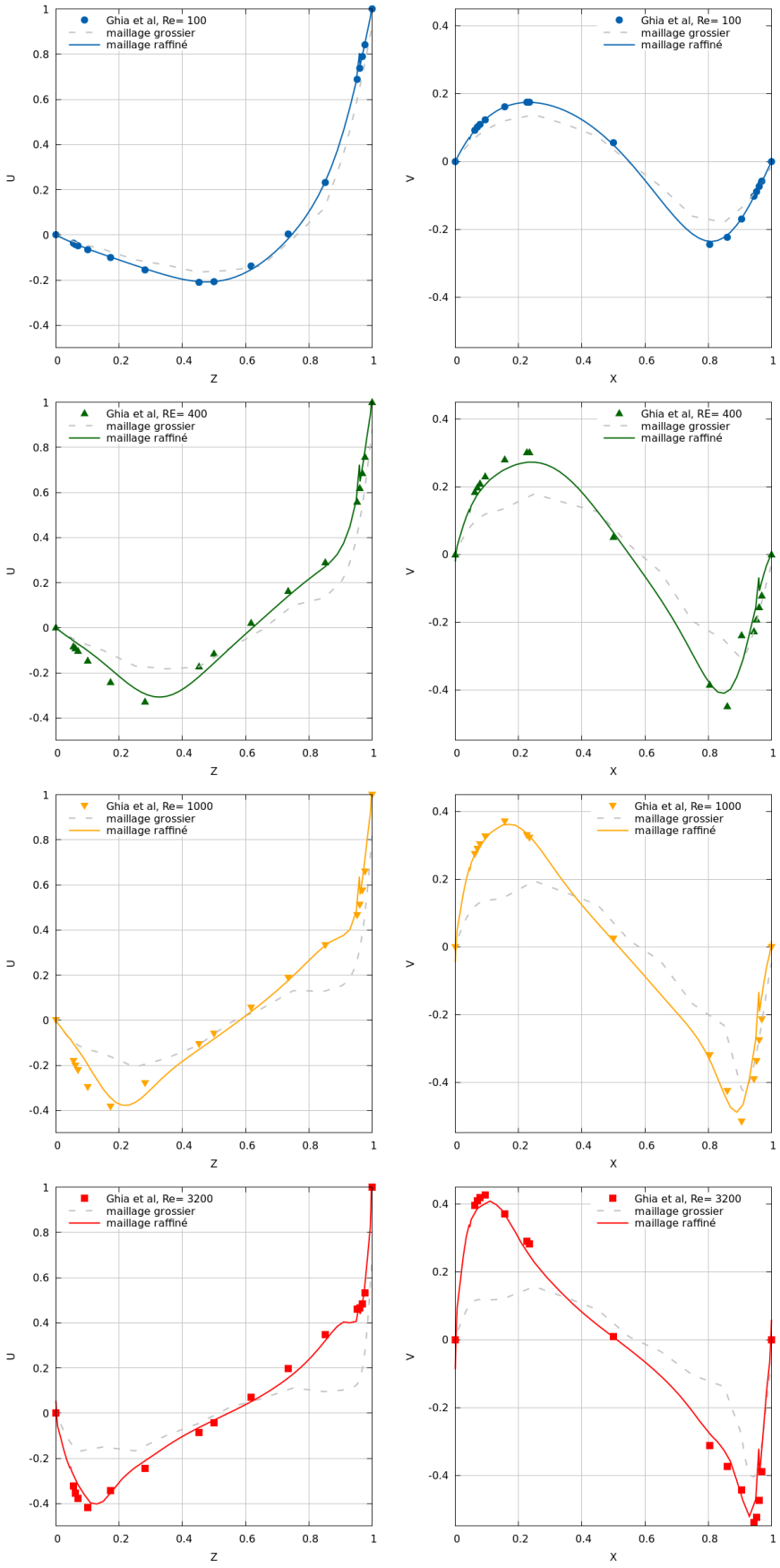


FIGURE 8.6 – Cavité entrainée : comparaison des vitesses horizontale u et verticale v respectivement sur les axes $x = 0.5$ et $z = 0.5$ sur le maillage raffiné

Chapitre 9

Écoulement laminaire 2D autour d'un obstacle carré fixe

9.1 Description

Un barreau de section carrée de côté $D = 1$ est plongé dans un écoulement plan allant de gauche à droite. La partie gauche du domaine constitue l'entrée du fluide où la vitesse est imposée égale à :

$$\begin{pmatrix} u \\ v \end{pmatrix}_{gauche} = \begin{pmatrix} U_\infty \\ 0 \end{pmatrix} \tag{9.1}$$

avec $U_\infty = 1\ m.s^{-1}$. Les autres bords du domaine sont des sorties et aucune symétrie n'est utilisée. Le centre du barreau est fixé à la position $(0, 0)$. Le domaine est construit de manière à pouvoir suivre les formations tourbillonnaires en aval de l'écoulement avec une longueur $10D$ en aval et de $15D$ en aval.

Le maillage initial considéré est un maillage non raffiné IBM. La géométrie de l'obstacle étant définie par un carré ayant les côtés paralleles aux axes, la paroi maillée ne souffre donc pas de crénelage caractéristique des maillages IBM sur des surfaces courbes. Ce cas test permettra de valider la prise en charge des maillages IBM par notre méthode de raffinement.

Les simulations sont réalisées pour les Reynolds 200, 250, 300, 400 et 500 en initialisant le calcul sur un maillage volontairement grossier constitué par le maillage de fond coloré en bleu sur la figure 9.1 qui sera raffiné par la suite par un maillage chimère/IBM plus fin et d'un maillage chimère du sillage.

Le barreau cylindrique à section carré est un cas très étudié dans la littérature [3, 114, 36]. En effet les structures à base carré, rectangulaire ou assimilable sont communes : gratte-ciels ou bâtiments plus modestes, clôtures, pylônes ou poutres forment autant d'exemples courants d'utilisations de cette géométrie dans les ouvrages qui nous entourent. Dans la réalité, au-delà d'un Reynolds dit "critique", des recirculations et tour-

billons se forment dans le sillage de l'écoulement entraînant divers modes de vibrations de la structure. Ces vibrations de faible amplitude peuvent contribuer à la fatigue des matériaux et à leur rupture lorsque celles-ci entraînent des phénomènes de résonances [115]. D'où l'intérêt de résoudre les instationnarités du sillage.

Les résultats des études citées précédemment portent des conclusions similaires sur les caractéristiques des écoulements autours de pavés droits, résultats présentant quelques nuances en fonctions des maillages. Selon [116], la transition entre l'état stationnaire et instationnaire pour ce type d'écoulement s'opère autour de $Re \sim 50$.

9.2 Résultats

Pour ce cas test deux maillages raffinés sont considérés. Les deux maillages raffinés ont pour base un même maillage cartésien uniforme de 125x100 de 0.2 mailles divisé en 8 blocs pour permettre un calcul en parallèle.

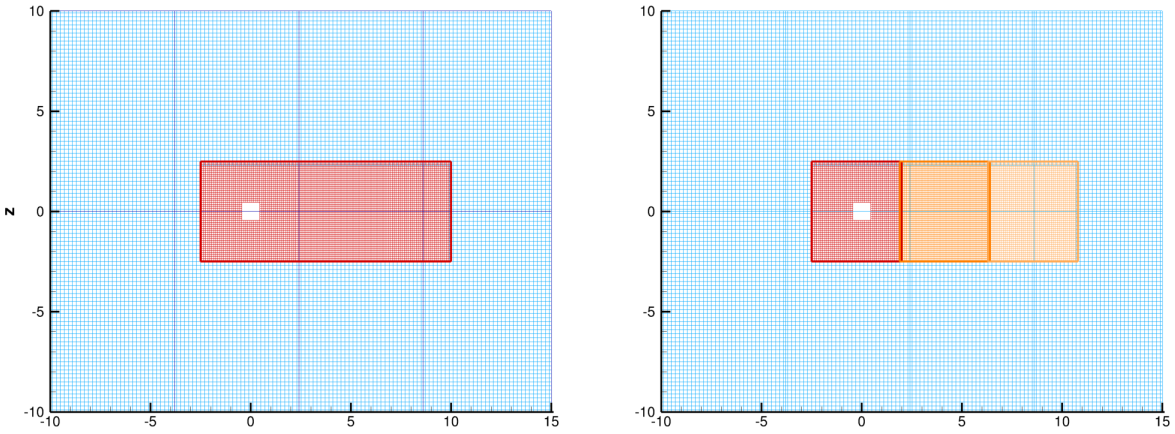


FIGURE 9.1 – Maillage initial constitué de 125x100 cellules en bleu. Le bloc chimère pour le raffinement simple à gauche est constitué de 125x50 cellules. Les blocs chimères pour le raffinement successif à droite sont constitués de 45x50 cellules.

Le premier maillage est raffiné à l'aide d'une seule grille supplémentaire autour du barreau central et permettra de valider la prise de la méthode pour l'ajout d'un bloc raffiné sur un calcul effectué en parallèle. Cette grille est cartésienne uniforme constituée de 125x50 cellules de 0.1. L'origine est prise à $x = -2.5$, $y = -2.5$.

Le second maillage est raffiné à l'aide de 3 grilles supplémentaires ajoutées successivement où chaque nouveau bloc est géré par un processeur distinct. Chaque bloc de maillage est constitué de 45x50 cellules de 0.1 chevauchant le bloc voisin de 0.1. Ce dernier cas test permettra de valider la méthode pour un calcul parallèle avec ajout de blocs successifs

gérés par différents processeurs. L'origine est prise à $x = -2.5$, $y = -2.5$.

Les simulations sont effectuées sur 8 processeurs en instationnaire, et présentées $t = 75s$ après initialisation sur le maillage grossier.

La Table 9.1 présente le nombre de Strouhal St ainsi que le coefficient de traînée moyen obtenus sur les maillages raffinés. Ces résultats sont comparés à ceux donnés dans les articles [114, 38]. Le coefficient de traînée est parfaitement en accord avec les résultats de la littérature. Le nombre de Strouhal s'écarte des expériences [114] mais est en accord avec les résultats obtenus sur un maillage de précision similaire généré par [38]. Cette différence peut être imputée notamment au maillage proche paroi du cylindre qui peut encore être raffiné d'avantage.

Source	Re	St	Cx moyen
Raffinement NSMB	200	0.156	1.46
NSIBM [38]	200	0.155	1.47
Sohankar, Norberg et Davidson [114]	200	0.170	1.46

TABLE 9.1 – Comparaison à Reynolds 200

Les figures 9.2, 9.3, 9.4, 9.5, présentent respectivement les champs de vitesse et de vorticité obtenus pour les Reynolds 200, 250, 300 et 400. Les écoulements sont identiques pour les deux types de maillages générés validant ainsi la prise en compte parallélisée du raffinement.

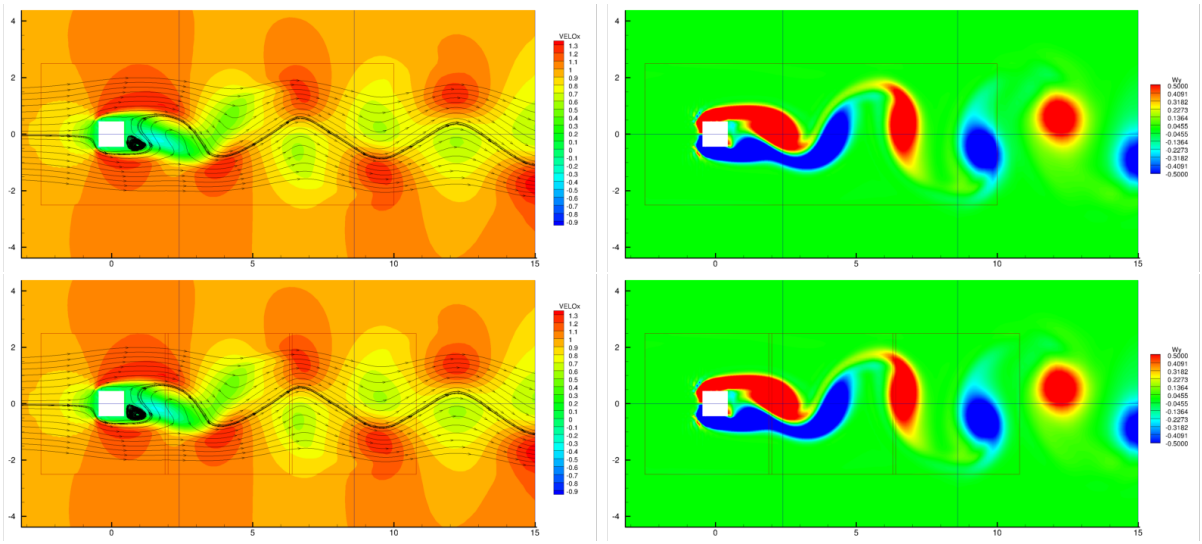


FIGURE 9.2 – Comparaison à Reynolds 200 du champs de vitesse et de la vorticité pour les deux raffinements adoptés

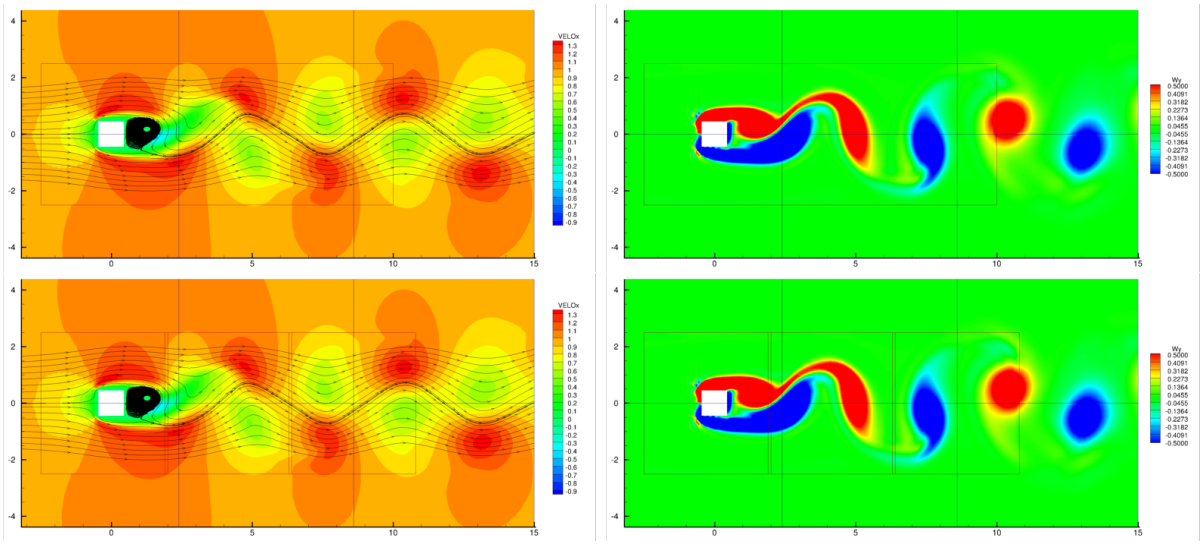


FIGURE 9.3 – Comparaison à Reynolds 250 du champs de vitesse et de la vorticité pour les deux raffinements adoptés

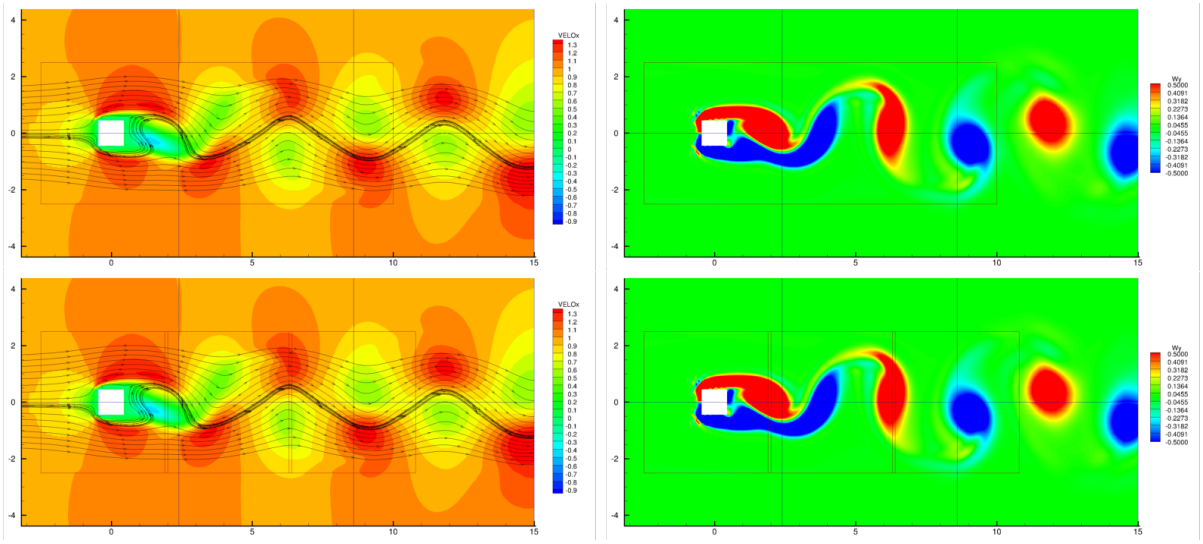


FIGURE 9.4 – Comparaison à Reynolds 300 du champs de vitesse et de la vorticité pour les deux raffinement adoptés

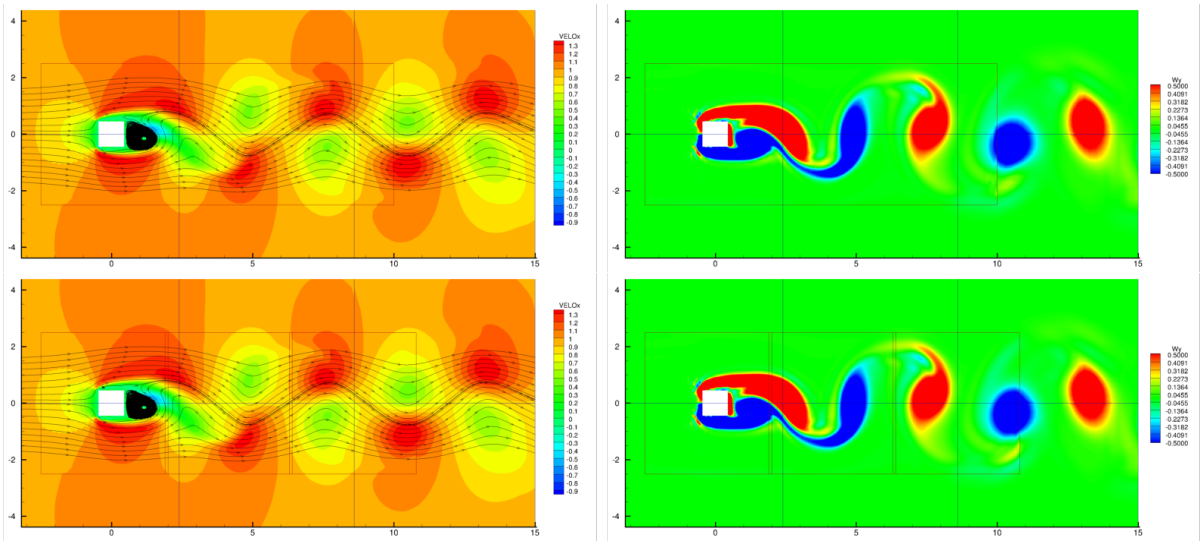


FIGURE 9.5 – Comparaison à Reynolds 400 du champs de vitesse et de la vorticité pour les deux raffinement adoptés

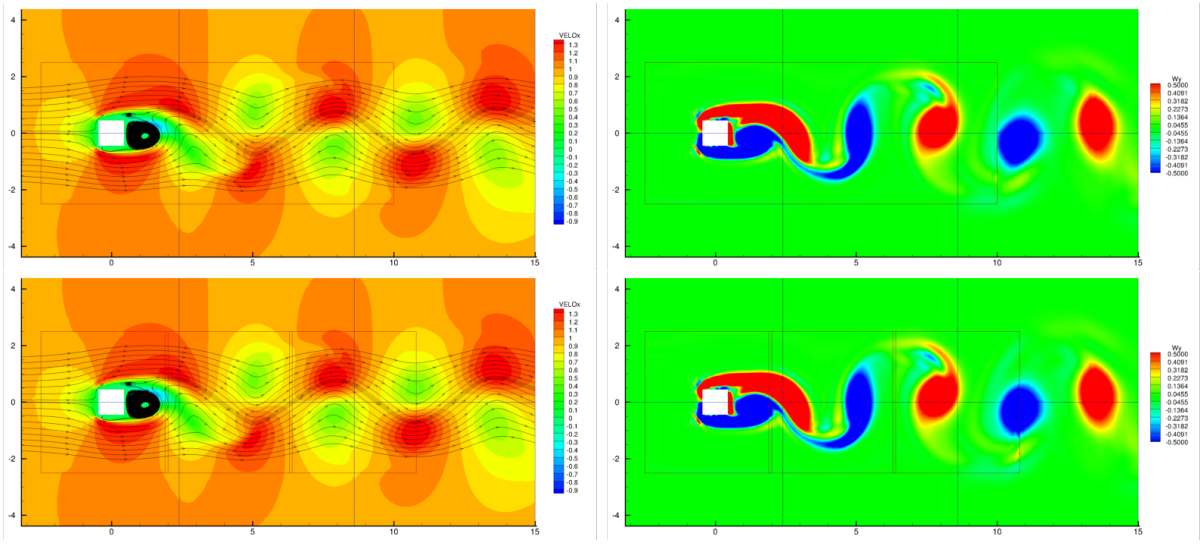


FIGURE 9.6 – Comparaison à Reynolds 500 du champs de vitesse et de la vorticité pour les deux raffinement adoptés

Chapitre 10

Écoulement laminaire 2D autour d'un obstacle circulaire fixe

10.1 Description

Un barreau cylindrique a section ronde de diamètre $D = 1$ est plongé dans un écoulement plan s'écoulant de gauche à droite. Les paramètres d'entrée sortie sont similaires au cas précédent du barreau à section carré. La partie gauche du domaine constitue l'entrée du fluide où la vitesse est imposée à :

$$\begin{pmatrix} u \\ v \end{pmatrix}_{gauche} = \begin{pmatrix} U_\infty \\ 0 \end{pmatrix} \tag{10.1}$$

avec $U_\infty = 1\ m.s^{-1}$. Les autres bords du domaine sont des sorties et aucune symétrie n'est utilisée. Le centre du barreau est fixé à la position $(0, 0)$. Le domaine est construit de manière à pouvoir suivre les formations tourbillonnaires en aval de l'écoulement avec $10D$ en aval, $15D$ en aval.

Dans la littérature il est classiquement admis que ce type d'écoulement est stationnaire pour des $Re < 50$. Pour des nombres de Reynolds plus élevés une allée de type Von Karman se forme dans le sillage du cylindre.

10.2 Résultats

10.2.1 État stationnaire

Plusieurs simulations sont réalisées pour des écoulements stationnaires à faibles Reynolds 10, 20, 30 et 40. Les résultats concernant la longueur de recirculation, l'angle de décollement et le coefficient de traînée obtenus à l'aide de notre raffinement codé dans *NSMB* sont comparés à la littérature existante et reportés sur les figures 10.2, 10.3 et 10.4. La figure 10.5 illustre l'évolution de la zone de recirculation de l'état stationnaire en fonction du nombre de Reynolds.

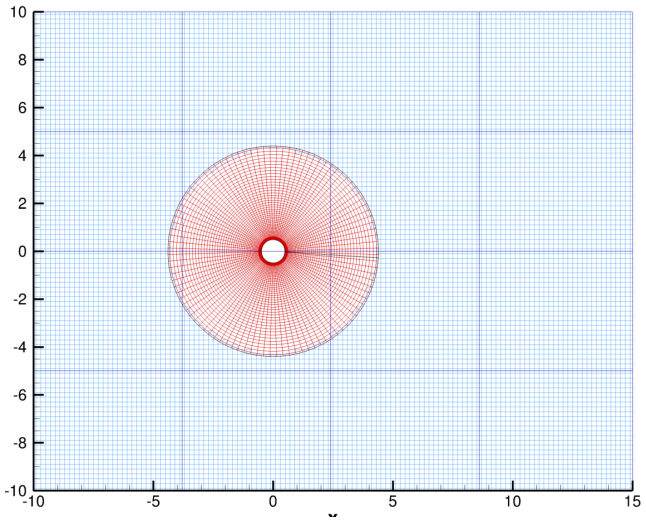


FIGURE 10.1 – Maillage résultant du raffinement chimère pour le barreau cylindrique fixe à section circulaire utilisé dans les régimes stationnaires

Le maillage de fond initial est identique au cas précédent du barreau à section carre. Cependant du fait de la courbure de l'obstacle nous optons ici pour un raffinement construisant un maillage conforme à la paroi. Le maillage résultant du raffinement est exposé en figure 10.1. Ce maillage est constitué de 101 cellules dans la direction parallèle à la paroi et de 58 cellules dans la direction radiale. Ces dernières 58 cellules son réparties en 2 cellules intérieures au cylindre et 15 cellules extérieures de hauteur 0.0075 suivies de 41 cellules de 0.15 de hauteur.

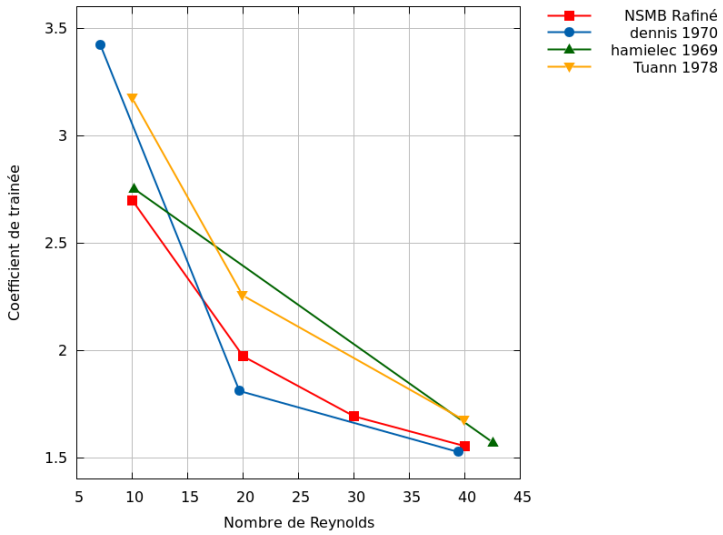


FIGURE 10.2 – Cylindre 2D : comparaison des coefficients de traînée. Les résultats obtenus avec le raffinement sont comparés aux autres résultats numériques obtenus par NSMB et la méthode chimère et ceux extraits de [**1970numerical**], [54] et [128].

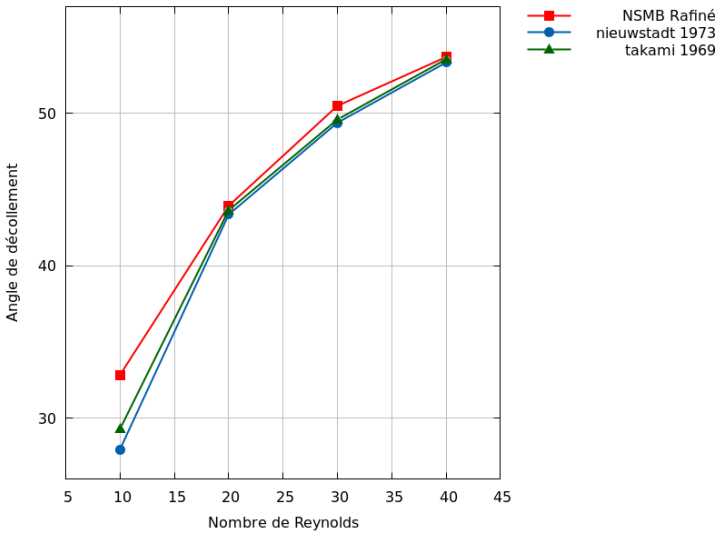


FIGURE 10.3 – Cylindre 2D : comparaison des angles de décollement . Les résultats obtenus avec le raffinement sont comparés aux autres résultats numériques obtenus par NSMB et la méthode chimère et ceux extraits de [95] et [121].

Les résultats sont en accords avec la littérature, permettant ainsi de valider le comportement de la méthode pour un écoulement bidimensionnel stationnaire avec maillage raffiné aux parois simulées par la méthode des frontières immergées.

10.2.2 État instationnaire

L'état instationnaire est simulé pour les Reynolds 100, 200 et 300. Dans cette plage de valeurs du nombre de Reynolds, l'écoulement instationnaire présente une allée tourbillonnaire périodique de Von Karman dans le sillage du cylindre caractérisé par le nombre adimensionnel de Strouhal :

$$St = \frac{fD}{U_\infty} \tag{10.2}$$

avec

- f la fréquence du lâché tourbillonnaire ,
- D la longueur caractéristique du problème (le diamètre du cylindre)
- U_∞ la vitesse caractéristique du problème (la vitesse imposée en entrée)

Le maillage de fond est identique au précédent. Du fait du lâché tourbillonnaire, le maillage est également raffiné dans le sillage du cylindre comme détaillé sur la figure 10.6. Le maillage autour du cylindre est constitué de 101 cellules dans la direction parallèle à la paroi et de 61 cellules dans la direction radiale. Ces dernières 61 cellules son réparties en 2 cellules intérieures au cylindre et 15 cellules extérieures de hauteur 0.0075 suivies de 44 cellules de 0.15 de hauteur. Les blocs raffinant le sillage sont des maillages cartésiens uniformes de 40x50 cellules de 0.1. l'origine est placée à $x = 2$ $y = -2.5$ et les blocs se

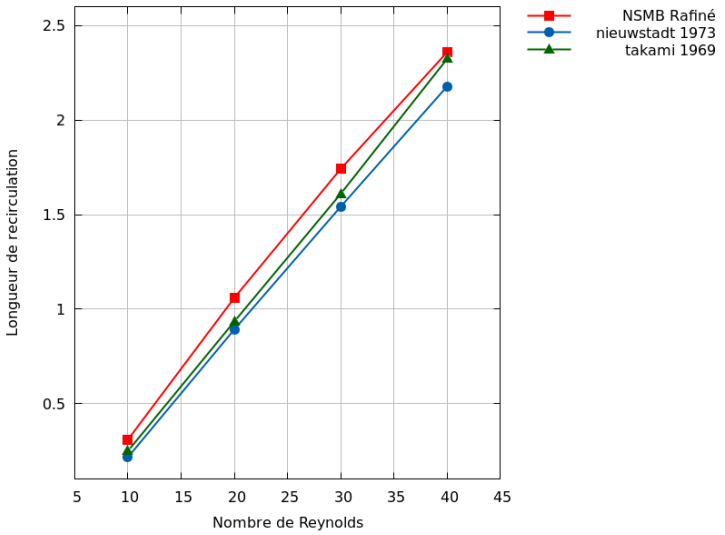


FIGURE 10.4 – Cylindre 2D : comparaison des longueurs de recirculation. Les résultats obtenus avec le raffinement sont comparés aux autres résultats numériques obtenus par NSMB et la méthode chimère et ceux extraits de [95] et [121].

chevauchent sur 0.3.

La figure 10.7 compare les résultats obtenus sur le maillage raffiné avec les données expérimentales extraites de l'article expérimentale [137] concernant un cylindre de très grande longueur de section circulaire ainsi que des données des simulations pour un cylindre 2D extraites de [100]. Les résultats obtenus sur le maillage raffiné sont en accords avec ceux de la littérature.

Les figures 10.8 et 10.9 illustrent respectivement les vitesses longitudinales et transverses obtenues pour les écoulements à Reynolds 100, 200 et 300.

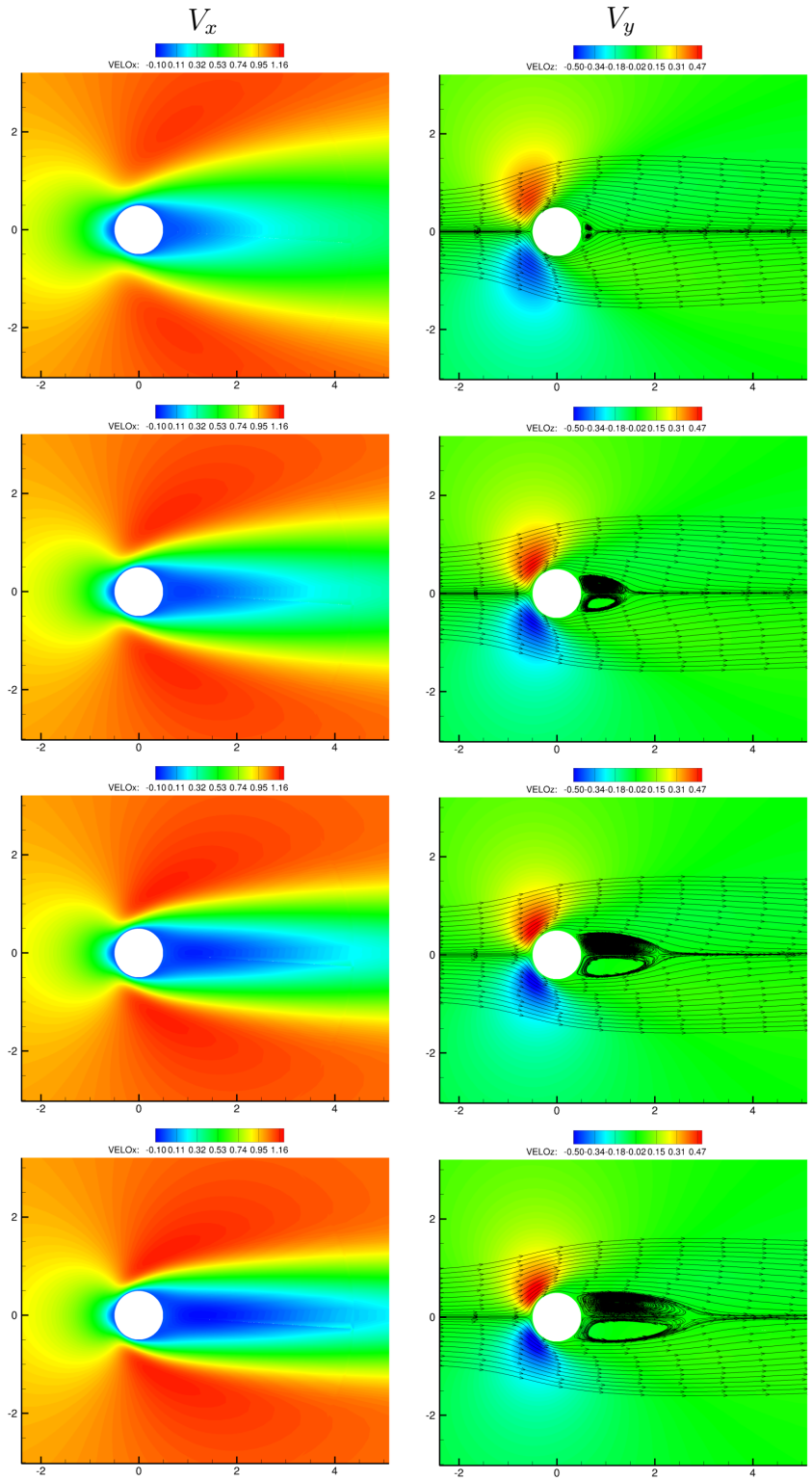


FIGURE 10.5 – Visualisation des champs de vitesses u et v pour les Reynolds - de haut en bas - 10, 20, 30 et 40

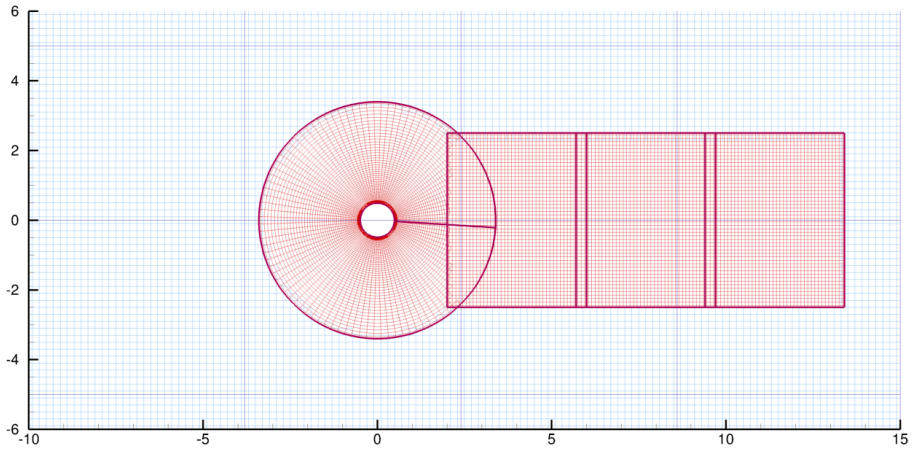


FIGURE 10.6 – Maillage résultant du raffinement pour le barreau cylindrique fixe à section ronde en configuration instationnaire

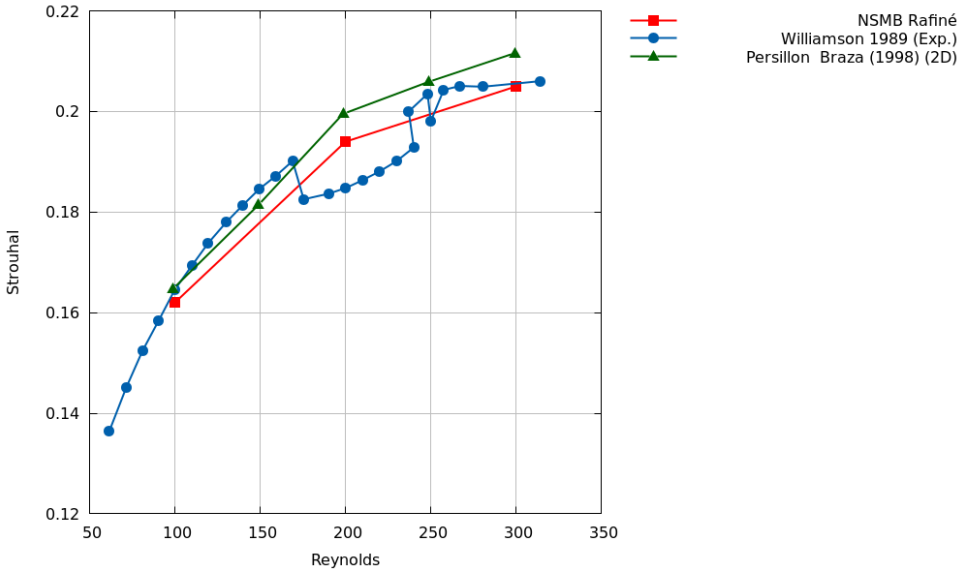
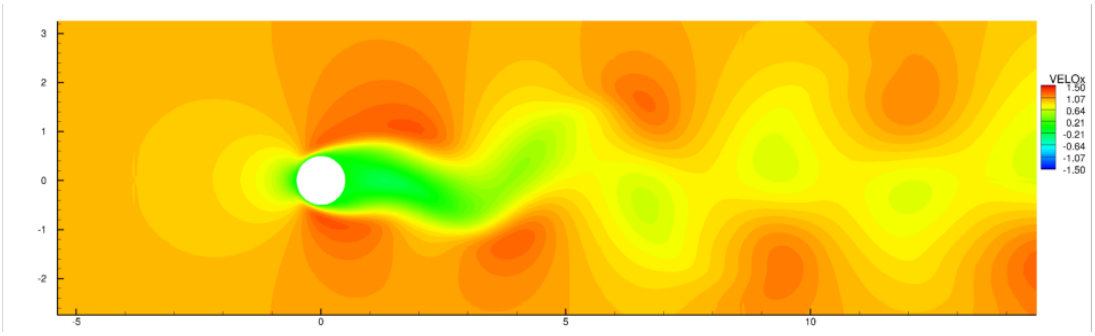
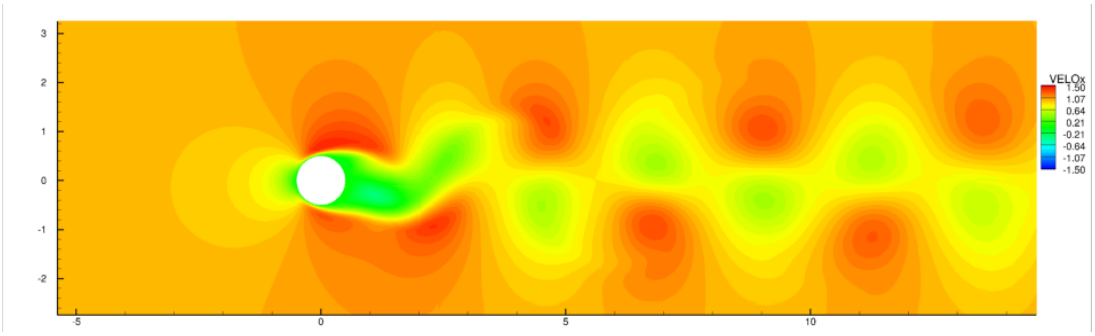


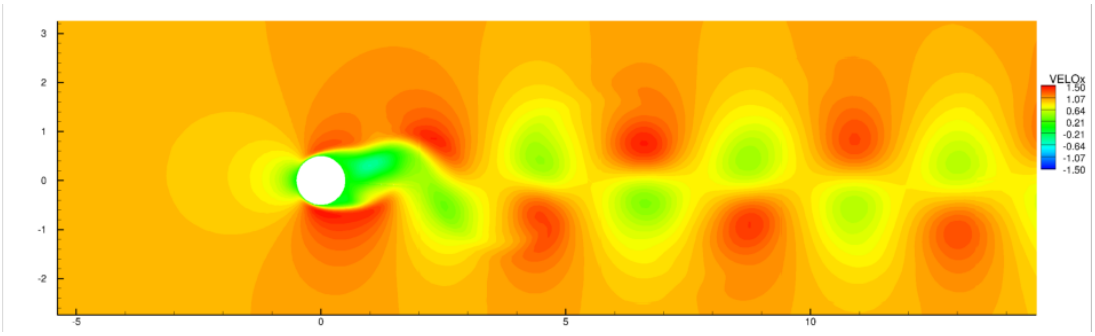
FIGURE 10.7 – Cylindre circulaire 2D : comparaison des nombres de Strouhal. Les résultats obtenus notre maillage raffiné sont comparés aux autres résultats numériques extraits de [137] et [100].



u à Re 100

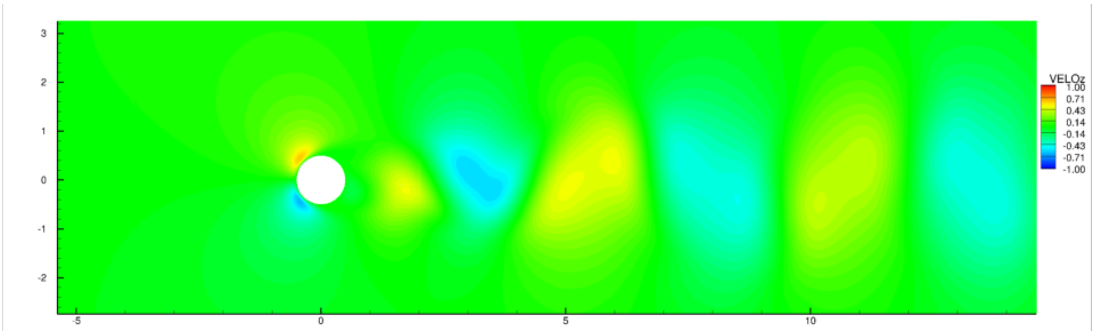


u à Re 200

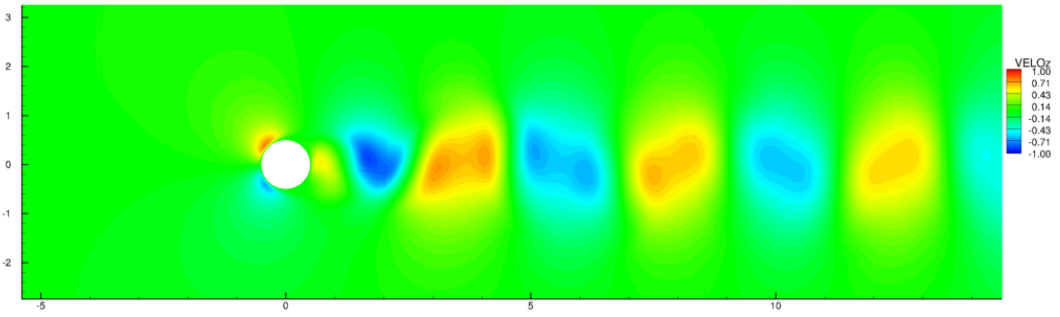


u à Re 300

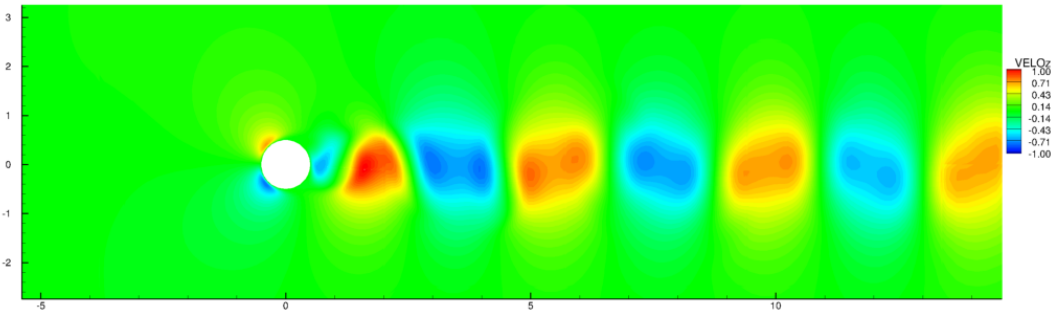
FIGURE 10.8 – État instationnaire du barreau cylindrique 2D : champs de vitesse horizontale u



v à Re 100



v à Re 200



v à Re 300

FIGURE 10.9 – État instationnaire du barreau cylindrique 2D : champ de vitesse verticale v

Quatrième partie

Conclusion

Ces travaux de thèse menés au sein du groupe *Instabilités, Turbulence, Multiphasique* de l'équipe MécaFlu du laboratoire ICube, département de Mécanique contribuent à étoffer le méthode Chimère du code *NSMB* de la fonction de raffinement à la volée, semi-automatique, 2D, parallélisé basé sur la méthode Chimère.

Dans un premier temps une modification de la structure de données du solveur *NSMB* a été réalisée afin de l'adapter à l'ajout de blocs de maillages au cours de l'exécution du calcul. Puis nous avons intégré la méthode de raffinement au sein de *NSMB*. Un ensemble d'options utilisateurs permettent de définir les paramètres du raffinement. Le code est fonctionnel avec l'utilisation conjointe de la méthode Chimère et de la méthode des frontières immergées.

Les résultats de notre méthode de raffinement ont été comparés sur un ensemble de cas bi-dimensionnels tirés de la littérature : cavité entrainée 2D, écoulement autour d'un barreau cylindriquei de section carrée en 2D et écoulement autour d'un barreau de section circulaire en 2D.

En perspective, le code nécessite des optimisations, l'intégration de critères de raffinement automatique, la prise en compte de la gestion d'ajouts de parois en frontière de maillage, la répartition de charge dynamique, la suppression de blocs et la modulation dynamique des communications chimères afin de développer son plein potentiel. Une validation étendue sur l'ensemble de la base de données de benchmark de *NSMB* doit également être prévue.

Bibliographie

[1] C.S. Peskin, « Flow patterns around heart valves : a digital computer method for solving the equations of motion », thèse de doct., Albert Einstein College of Medicine of Yeshiva University, 1972.

[2] M. J. Aftosmis, « Solution adaptive cartesian grid methods for aerodynamic flows with complex geometries », *in* : *VKI Lecture Series* 2 (1997), p. 1–105.

[3] M. Sukri M. Ali, C. J. Doolan et V. Wheatley, « Grid convergence study for a two-dimensional simulation of flow around a square cylinder at a low reynolds number », *in* : *Seventh International Conference on CFD in The Minerals and Process Industries, CSIRO (CSIRO Australia, Melbourne, Australia, 2009)*, 2009.

[4] P. Angot, C. H. Bruneau et P. Fabbrie, « A penalization method to take into account obstacles in viscous flows », *in* : *Numerische Mathematik* 81 (1999), p. 139–65.

[5] O. Antepara et al., « Parallel adaptive mesh refinement for large-eddy simulations of turbulent flows », *in* : *Computers & Fluids* 110 (2015), p. 48–61.

[6] Ivo Babuska et Barna Szabo, « On the rates of convergence of the finite element method », *in* : *International Journal for Numerical Methods in Engineering* 18.*3* (1982), p. 323–341.

[7] Ivo Babuska, Barna A Szabo et I Norman Katz, « The p-version of the finite element method », *in* : *SIAM journal on numerical analysis* 18.*3* (1981), p. 515–545.

[8] L Babuška et Manil Suri, « The optimal convergence rate of the p-version of the finite element method », *in* : *SIAM Journal on Numerical Analysis* 24.*4* (1987), p. 750–776.

[9] Timothy J Baker, « Mesh adaptation strategies for problems in fluid dynamics », *in* : *Finite Elements in Analysis and Design* 25.*3-4* (1997), p. 243–273.

[10] E. Balaras, « Modeling complex boundaries using an external force field on fixed Cartesian grids in large-eddy simulations », *in* : *Computers and Fluids* 33 (2004), p. 375–404.

[11] Gang Bao, Guanghui Hu et Di Liu, « An h-adaptive finite element solver for the calculations of the electronic structures », *in* : *Journal of Computational Physics* 231.*14* (2012), p. 4967–4979.

[12] FB Barros, SPB Proença et CS de Barcellos, « Generalized finite element method in structural nonlinear analysis–a p-adaptive strategy », *in* : *Computational Mechanics* 33.*2* (2004), p. 95–107.

[13] Prodyot K Basu et Alberto Peano, « Adaptivity in p-version finite element analysis », *in* : *Journal of Structural Engineering* 109.*10* (1983), p. 2310–2324.

[14] J. D. Baum, H. Luo et R. Loehner, « The numerical simulation of strongly unsteady flows with hundreds of moving bodies », *in* : *36th AIAA Aerospace Sciences Meeting and Exhibit*, t. 788, 1998.

[15] Ted Belytschko et M Tabbara, « H-adaptive finite element methods for dynamic problems, with emphasis on localization », *in* : *International Journal for Numerical Methods in Engineering* 36.*24* (1993), p. 4245–4265.

[16] J Benek, J Steger et F Carroll Dougherty, « A flexible grid embedding technique with application to the Euler equations », *in* : *6th Computational Fluid Dynamics Conference Danvers*, 1983, p. 1944.

[17] Marsha J Berger et Phillip Colella, « Local adaptive mesh refinement for shock hydrodynamics », *in* : *Journal of computational Physics* 82.*1* (1989), p. 64–84.

[18] M.J. Berger et M.J. Aftosmis, « Aspects (and aspect ratios) of Cartesian mesh methods », *in* : *Proc. 16th Int. Conf. Numer. Methods Fluid Dyn*, 1998.

[19] R. P. Beyer et R. J. Leveque, « An adaptive Cartesian mesh algorithm for the euler equations in arbitrary geometries. », *in* : *9TH COMPUTATIONAL FLUID DYNAMICS CONFERENCE*, 1989.

[20] R. P. Beyer et R. J. Leveque, « Analysis of a one-dimensional model for the immersed boundary method », *in* : *SIAM Journal on Numerical Analysis* 29 (1992), p. 332–64.

[21] R.P. Beyer, « A computational model of the cochlea using the immersed boundary method », *in* : *Journal of Computational Physics* 98 (1992), p. 145–62.

[22] Achi Brandt, « Multi-level adaptive solutions to boundary-value problems », *in* : *Mathematics of computation* 31.*138* (1977), p. 333–390.

[23] H. C. Brinkmann, « A calculation of the viscous force exerted by a flowing fluid on a swarm of particles », *in* : *Applied Science Research* 1 (1947), p. 27–34.

[24] PG Buning et al., « Numerical simulation of the integrated space shuttle vehicle in ascent », *in* : *AIAA Atmospheric Flight Mechanics Conference, Minneapolis, MN*, 1988, p. 265–283.

[25] T.M. Burton et J.K. Eaton, « Analysis of a fractional-step method on overset grids », *in* : *Journal of Computational Physics* 177 (2002), p. 336–364.

[26] Zhenning Cai et Ruo Li, « An h-adaptive mesh method for Boltzmann-BGK/hydrodynamics coupling », *in* : *Journal of Computational Physics* 229.*5* (2010), p. 1661–1680.

[27] Jean-Jacques Chattot et Y. Wang, « Improved treatment of intersecting bodies with the chimera method and validation with a simple and fast flow solver », *in* : *Computers & fluids* 27 (1998), p. 721–740.

[28] G. Chesshire et WD Henshaw, « Composite overlapping meshes for the solution of partial differential equations », *in* : *Journal of Computational Physics* 90 (1990), p. 1–64.

[29] Alexandre Joel Chorin, « Numerical solution of the Navier-Stokes equations », *in* : *Mathematics of computation* 22.*104* (1968), p. 745–762.

[30] W. J. Coirier et K. G. Powell, « Solution-adaptive Cartesian cell approach for viscous and inviscid flows », *in* : *AIAA Journal* 34.*5* (1996), p. 938–45.

[31] Thibaut DELOZE, « Couplage fluide-solide appliqué à l'étude de mouvement d'une sphère libre dans un tube vertical », thèse de doct., Université de Strasbourg, 2011.

[32] Thibaut Deloze, *NSMB consortium annual meeting*, 2013.

[33] L Demkowicz, Ph Devloo et J Tinsley Oden, « On an h-type mesh-refinement strategy based on minimization of interpolation errors », *in* : *Computer Methods in Applied Mechanics and Engineering* 53.*1* (1985), p. 67–89.

[34] G. Desquesnes et al., « On the use of a high order overlapping grid method for coupling in CFD/CAA », *in* : *Journal of Computational Physics* 220 (2006), p. 355–382.

[35] H Ding et al., « Development of least-square-based two-dimensional finite-difference schemes and their application to simulate natural convection in a cavity », *in* : *Computers & Fluids* 33.*1* (2004), p. 137–154.

[36] C. Doolan, *in* : *AIAA journal* 47.*2* (2009), p. 475–479.

[37] Anshu Dubey et al., « A survey of high level frameworks in block-structured adaptive mesh refinement packages », *in* : *Journal of Parallel and Distributed Computing* 74.*12* (2014), p. 3217–3227, URL : `http://dx.doi.org/10.1016/j.jpdc.2014.07.001`.

[38] Daniel Durrenberger, « NSIBM : un solveur parallèle de Navier-Stokes avec raffinement automatique basé sur la méthode des frontières immergées », 2015STRAD049, thèse de doct., 2015, URL : `http://www.theses.fr/2015STRAD049/document`.

[39] Alexander Düster et Ernst Rank, « The p-version of the finite element method compared to an adaptive h-version for the deformation theory of plasticity », *in* : *Computer Methods in Applied Mechanics and Engineering* 190.*15-17* (2001), p. 1925–1935.

[40] Wolfgang Ehlers, Martin Ammann et Stefan Diebels, « h-Adaptive FE methods applied to single-and multiphase problems », *in* : *International journal for numerical methods in engineering* 54.*2* (2002), p. 219–239.

[41] EA Fadlun et al., « Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations », *in* : *Journal of Computational Physics* 161.*1* (2000), p. 35–60.

[42] L.J. Fauci et A. McDonald, « Sperm motility in the presence of boundaries », *in* : *Bulletin of Mathematical Biology* 57 (1994), p. 679–99.

[43] J.H. Ferzinger et M. Perić, *Computational Methods for Fluid Dynamics*, Springer, 1998, isbn : 9783642976513, url : `http://books.google.fr/books?id=mbd3NAEACAAJ`.

[44] Jacob Fish et Stilianos Markolefas, « Adaptive global–local refinement strategy based on the interior error estimates of the h-method », *in* : *International Journal for Numerical Methods in Engineering* 37.*5* (1994), p. 827–838.

[45] *fortran tool*, 2012, url : `https://github.com/arktools/fortran_tools`.

[46] K Fujii, « Unified Zonal Method on the Fortified Solution Algorithm », *in* : *J. Comput. Phys.* 118 (1995), p. 92–108.

[47] U Ghia, K.N Ghia et C.T Shin, « High-Re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method », *in* : *Journal of Computational Physics* 48.*3* (1982), p. 387–411, issn : 0021-9991, doi : `https://doi.org/10.1016/0021-9991(82)90058-4`, url : `http://www.sciencedirect.com/science/article/pii/0021999182900584`.

[48] U. Ghia, KN Ghia et CT Shin, « High-Re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method », *in* : *Journal of Computational Physics* 48 (1982), p. 387–411.

[49] Reza Ghias, Rajat Mittal et Thomas S Lund, « A non-body conformal grid method for simulation of compressible flows with complex immersed boundaries », *in* : *AIAA paper* 80 (2004), p. 2004.

[50] A Gilmanov, F Sotiropoulos et E Balaras, « A general reconstruction algorithm for simulating flows with complex 3D immersed boundaries on Cartesian grids », *in* : *Journal of Computational Physics* 191.*2* (2003), p. 660–669.

[51] Anvar Gilmanov et Fotis Sotiropoulos, « A hybrid Cartesian/immersed boundary method for simulating flows with 3D, geometrically complex, moving bodies », *in* : *Journal of Computational Physics* 207.*2* (2005), p. 457–492.

[52] PA Gnoffo, « A finite-volume, adaptive grid algorithm applied to planetary entry flowfields », *in* : *AIAA journal* 21.*9* (1983), p. 1249–1254.

[53] D. Goldstein, R. Handler et L. Sirovitch, « Modeling a no-slip flow boundary with an external force field », *in* : *Journal of Computational Physics* 105 (1993), p. 354–66.

[54] A.E. Hamielec et J.D. Raal, « Numerical studies of viscous flow around circular cylinders », *in* : *Physics of Fluids (1958-1988)* 12.*1* (1969), p. 11–17.

[55] Chung-Souk Han et Peter Wriggers, « An h-adaptive method for elasto-plastic shell problems », *in* : *Computer methods in applied mechanics and engineering* 189.*2* (2000), p. 651–671.

[56] O Hassan, EJ Probert et K Morgan, « Unstructured mesh procedures for the simulation of three-dimensional transient compressible inviscid flows with moving boundary components », *in* : *International journal for numerical methods in fluids* 27.*1-4* (1998), p. 41–55.

[57] A Hay et M Visonneau, « Computation of free-surface flows with local mesh adaptation », *in* : *International journal for numerical methods in fluids* 49.*7* (2005), p. 785–816.

[58] W.D. Henshaw, « A fourth-order accurate method for the incompressible Navier-Stokes equations on overlapping grids », *in* : *Journal of Computational Physics* 113 (1994), p. 13–25.

[59] William D Henshaw et Donald W Schwendeman, « An adaptive numerical scheme for high-speed reactive flow on overlapping grids », *in* : *Journal of Computational Physics* 191.*2* (2003), p. 420–447.

[60] William D Henshaw et Donald W Schwendeman, « Moving overlapping grids with adaptive mesh refinement for high-speed reactive and non-reactive flow », *in* : *Journal of Computational Physics* 216.*2* (2006), p. 744–779.

[61] B. Hubbard et H. C. Chen, « A chimera scheme for incompressible viscous flows with application to submarine hydrodynamics », *in* : *AIAA Paper, 25th AIAA Fluid Dynamics Conference* 2210 (1990).

[62] G. Iaccarino, G. Kalitzin et G. J. Elkins, *Numerical and experimental investigation of the turbulent flow in a ribbed serpentine passage*, rapp. tech., DTIC Document, 2003.

[63] Gianluca Iaccarino et Roberto Verzicco, « Immersed boundary technique for turbulent flow simulations », *in* : *Applied Mechanics Reviews* 56.*3* (2003), p. 331–347.

[64] P. Mornas J. Bohbot S. Champagneux, *NSMB 90, Document de spécifications/-conceptions*, rapp. tech., CERFACS, 2013.

[65] A. Jahangirian et M. Y. Hashemi, « Adaptive Cartesian grid with mesh-less zones for compressible flow calculations », *in* : *Computers & Fluids* 54 (2012), p. 10–17.

[66] Antony Jameson, « Analysis and design of numerical schemes for gas dynamics, 1 : artificial diffusion, upwind biasing, limiters and their effect on accuracy and multigrid convergence », *in* : *International Journal of Computational Fluid Dynamics* 4.*3-4* (1995), p. 171–218.

[67] D. Jespersen, TH Pulliam et P. Buning, « Recent enhancements to OVERFLOW », *in* : *AIAA paper* 97–0644 (1997).

[68] G. Kalitzin, G. Iaccarino et B. Khalighi, « Towards an immersed boundary solver for rans simulations », *in* : *AIAA Pap* 770.*2003* (2003), p. 33.

[69] AK Kapila et al., « A study of detonation diffraction in the ignition-and-growth model », *in* : *Combustion Theory and Modelling* 11.*5* (2007), p. 781–822.

[70] K. Khadra, P. Angot et S. Parneix, « Fictitious domain approach for numerical modeling of Navier-Stokes equations », *in* : *International Journal for Numerical Methods in Fluids* 34 (2000), p. 651–84.

[71] C Kiris et al., « Computational approach for probing the flow through artificial heart devices », *in* : *Journal of biomechanical engineering* 119.*4* (1997), p. 452–460.

[72] Ethan J Kubatko et al., « Dynamic p-adaptive Runge–Kutta discontinuous Galerkin methods for the shallow water equations », *in* : *Computer Methods in Applied Mechanics and Engineering* 198.*21-26* (2009), p. 1766–1774.

[73] Pierre Ladevèze et Jean-Pierre Pelle, *Mastering calculations in linear and nonlinear mechanics*, t. 171, Springer, 2005.

[74] Py LadeVeZe, J-P Pelle et Ph Rougeot, « Error estimation and mesh optimization for classical finite elements », *in* : *Engineering Computations* 8.*1* (1991), p. 69–80.

[75] M.C. Lai et C.S. Peskin, « An immersed bound- ary method with formal second-order accuracy and reduced numerical viscosity », *in* : *Journal of Computational Physics* 160 (2000), p. 709–19.

[76] B. Landmann et M. Montagnac, « A highly automated parallel Chimera method for overset grids based on the implicit hole cutting technique », *in* : *International Journal for Numerical Methods in Fluids* (2010), doi : 10.1002/fld.2292.

[77] Frédéric Lebon, Michel Raous et Iulian Rosu, « Multigrid methods for unilateral contact problems with friction », *in* : *IUTAM Symposium on Computational Methods in Contact Mechanics*, Springer, 2007, p. 1–16.

[78] L-Y Li et al., « Theoretical formulations for adaptive finite element computations », *in* : *Communications in Numerical Methods in Engineering* 11.*10* (1995), p. 857–868.

[79] W. Liao, J. Cai et H.M. Tsai, « A multigrid overset grid flow solver with implicit hole cutting method », *in* : *Computer Methods in Applied Mechanics and Engineering* 196 (2007), p. 1701–1715.

[80] A. Lintermann et al., « Massively parallel grid generation on HPC systems », *in* : *Compter Methods in Appplied Mechanics and Engineering* 277 (2014), p. 131–53.

[81] AA Lubrecht, WE Ten Napel et R Bosma, « Multigrid, an alternative method of solution for two-dimensional elastohydrodynamically lubricated point contact calculations », *in* : *Journal of tribology* 109.*3* (1987), p. 437–443.

[82] RH Magarvey et R.L. Bishop, « Transition ranges for three-dimensional wakes », *in* : *Canadian Journal of Physics* 39.*10* (1961), p. 1418–1422, issn : 1208-6045.

[83] S. Majumdar, G. Iaccarino et P.A. Durbin, « RANS solver with adaptive structured boundary non-conforming grids », *in* : *Annual Research Briefs, Center for Turbulence Research* (2001), p. 353–64.

[84] R. Maple et D. Belk, « A new approach to domain decomposition, the beggar code », *in* : *Numerical Grid Generation in Computational Fluid Dynamics and Related Fields, Pineridge Press Limited* (1994), p. 305–314.

[85] R. L. Meakin, *Chapter 11 : Composite overset structured grids, in : Handbook of grid generation, J.F. Thompson, B.K. Soni and N.P. Weatherill*, CRC, 1999, p. 1–20, isbn : 0849326877.

[86] R.L. Meakin et N.E. Suhs, « Unsteady aerodynamic simulation of multiple bodies in relative motion », *in* : *AIAA Computational Fluid Dynamics Conference, 9 th, Buffalo, NY*, t. AIAA-89-1996, 1989, p. 643–657.

[87] Mentor Graphics Whitepaper, *Advanced immersed boundary cartesian meshing technology in FloEFD.* www.mentor.com, 2011.

[88] C. Merlin, « Simulation numérique de la combustion turbulente : Méthode de frontères immergées pour les écoulements compressibles, application à la combustion en aval d'une cavité. », thèse de doct., INSA de Rouen, 2011.

[89] R. Mittal et G. Iaccarino, « Immersed boundary methods », *in* : *Annual Review of Fluid Mechanics* 37 (2005), p. 239–61, doi : 10.1146/annurev.fluid.37.061903.175743.

[90] R. Mittal, V. Seshadri et H.S. Udaykumar, « Flutter, tumble and vortex induced autorotation », *in* : *Theoretical and Computational Fluid Dynamics* 17.*3* (2004), p. 165–70.

[91] R. Mittal, Y. Utturkar et H. S. Udaykumar, « Computational modeling and analysis of biomimetic flight mechanisms », *in* : *AIAA Paper* 865 (2002), p. 2002.

[92] S. Mittal et B. Kumar, « Flow past a rotating cylinder », *in* : *Journal of Fluid Mechanics* 476 (2003), p. 303–334, issn : 0022-1120.

[93] J. Mohd-Yusof, « Combined immersed-boundary/B-spline methods for simulations of ow in complex geometries », *in* : *Center for Turbulence Research. Annual Research Briefs.* (1997), p. 317–327.

[94] K. Nakahashi et L. Kim, « Building-Cube Method for Large-Scale High Resolution Flow Computations », *in* : *42nd AIAA Aerospace Sciences Meeting and Exhibit*, Reno, Nevada, 2004.

[95] F. Nieuwstadt et H.B. Keller, « Viscous flow past circular cylinders », *in* : *Computers & Fluids* 1.*1* (1973), p. 59–71.

[96] B Palmerio, « A two-dimensional FEM adaptive moving-node method for steady Euler flow simulations », *in* : *Computer Methods in Applied Mechanics and Engineering* 71.*3* (1988), p. 315–340.

[97] B Palmerio, « An attraction-repulsion mesh adaption model for flow solution on unstructured grids », *in* : *Computers & fluids* 23.*3* (1994), p. 487–506.

[98] D.G. Pearce et al., « Development of a large scale Chimera grid system for the space shuttle launch vehicle », *in* : *AIAA, 31st Aerospace Sciences Meeting, Reno, NV*, t. 0533, 1993.

[99] Dorian Pena, « Développement d'un code de givrage tridimensionnel avec méthode Level-Set », 2016STRAD014, thèse de doct., 2016, url : http://www.theses.fr/2016STRAD014/document.

[100] H. Persillon et M. Braza, « Physical analysis of the transition to turbulence in the wake of a circular cylinder by three-dimensional Navier–Stokes simulation », *in* : *Journal of Fluid Mechanics* 365 (1998), p. 23–88.

[101] C.S. Peskin, « The fluid dynamics of heart valves : experimental, theoretical and computational methods », *in* : *Annual Review of Fluid Mechanics* 14 (1981), p. 235–59.

[102] Shahyar Pirzadeh, « An adaptive unstructured grid method by grid subdivision, local remeshing, and grid movement », *in* : *14th Computational Fluid Dynamics Conference*, 1999, p. 3255.

[103] K.G. Powell, P.L. Roe et J. Quirk, « Adaptive-mesh algorithms for computational fluid dynamics », *in* : *Algorithmic trends in computational fluid dynamics* 53 (1993), p. 303–37.

[104] A. Quarteroni et A. Valli, *Domain Decomposition Methods for Partial Differential Equations*, Numerical mathematics and scientific computation, Clarendon Press, 1999, isbn : 9780198501787, url : http://books.google.fr/books?id=CswwZ9Ca9IIC.

[105] W Rachowicz, « An anisotropic h-adaptive finite element method for compressible Navier-Stokes equations », *in* : *Computer Methods in Applied Mechanics and Engineering* 146.*3-4* (1997), p. 231–252.

[106] Man Mohan Rai, « A relaxation approach to patched-grid calculations with the euler equations », *in* : *Journal of Computational Physics* 66 (1986), p. 99–131, issn : 0021-9991, doi : DOI:10.1016/0021-9991(86)90056-2, url : http://www.sciencedirect.com/science/article/B6WHY-4DD1W2P-16X/2/45c1b27ce7fbec6ecd6d22afdfa23b2c.

[107] R. Ramamurti et W.C. Sandberg, « Simulation of flow about flapping airfoils using a finite element incompressible flow solver », *in* : *AIAA J.* 39 (2001), p. 253–352.

[108] C. Rome, « Une méthode de raccordement de maillages non-conformes pour la résolution des équations de Navier-Stokes », thèse de doct., Université Sciences et Technologies - Bordeaux I, 2006.

[109] E.M. Saiki et S. Biringen, « Numerical simulation of a cylinder in uniform flow : application of a virtual boundary method », *in* : *Journal of Computational Physics* 123 (1996), p. 450–65.

[110] H. Samet, *The design and analysis of spatial data structures*, Addison-Wesley, 1990.

[111] T.K. Sengupta, VK Suman et N. Singh, « Solving Navier–Stokes equation for flow past cylinders using single-block structured and overset grids », *in* : *Journal of Computational Physics* 229.*1* (2010), p. 178–199, issn : 0021-9991.

[112] S.E. Sherer et J.N. Scott, « High-order compact finite-difference methods on general overset grids », *in* : *Journal of Computational Physics* 210 (2005), p. 459–496.

[113] Timo L. Siikonen, Patrick P. Rautaheimo et Esa J. Salminen, « Numerical techniques for complex aeronautical flows », *in* : *European Congress on Computational Methods in Applied Sciences and Engineering ECCOMAS 2000* (2000).

[114] A. Sohankar, C. Norberg et L. Davidson, « Low-Reynolds-number flow around a square cylinder at incidence : study of blockage, onset of vortex shedding and outlet boundary condition », *in* : *International journal for numerical methods in fluids* 26.*1* (1998), p. 39–56.

[115] A. Sohankar, C. Norberg et L. Davidson, « Numerical simulation of flow past a square cylinder », *in* : *Proceedings of FEDSM99 3rd ASME/JSME Joint Fluids Engineering Conference. July*, 1999, p. 18–23.

[116] A. Sohankar, C. Norberg et L. Davidson, « Numerical simulation of unsteady low-Reynolds number flow around rectangular cylinders at incidence », *in* : *Journal of Wind Engineering and Industrial Aerodynamics* 69 (1997), p. 189–201.

[117] J.M. Stockie et B.R. Wetton, « Analysis of stiffness in the immersed boundary method and implications for time-stepping schemes », *in* : *Journal of Computational Physics* 154 (1998), p. 41–64.

[118] T Strouboulis et KA Haque, « Recent experiences with error estimation and adaptivity, Part II : Error estimation for h-adaptive approximations on grids of triangles and quadrilaterals », *in* : *Computer Methods in Applied Mechanics and Engineering* 100.*3* (1992), p. 359–430.

[119] Y. Tahara et al., « RANS simulation of a container ship using a single-phase level-set method with overset grids and the prognosis for extension to a self-propulsion simulator », *in* : *Journal of Marine Science and Technology* 11.*4* (2006), p. 209–228, issn : 0948-4280.

[120] CH Tai, Y. Zhao et KM Liew, « Parallel computation of unsteady incompressible viscous flows around moving rigid bodies using an immersed object method with overlapping grids », *in* : *Journal of Computational Physics* 207 (2005), p. 151–172.

[121] H. Takami et H. B. Keller, « Steady Two-Dimensional Viscous Flow of an Incompressible Fluid past a Circular Cylinder », *in* : *Physics of Fluids (1958-1988)* 12.*12* (1969), p. II–51.

[122] HS Tang, S. Casey Jones et F. Sotiropoulos, « An overset-grid method for 3D unsteady incompressible flows », *in* : *Journal of Computational Physics* 191 (2003), p. 567–600.

[123] RT Tenchev et al., « Finite element moving mesh analysis of phase change problems with natural convection », *in* : *International journal of heat and fluid flow* 26.*4* (2005), p. 597–612.

[124] T.E. Tezduyar, « Finite element methods for flow problems with moving boundaries and interfaces », *in* : *Archives of Computational Methods in Engineering* 8 (2001), p. 83–130.

[125] J.F. Thompson, F.C. Thomas et C.W. Mastin, « Automated numerical generation of body fitted curvilinear coordinate system for field containing any number of arbitrary 2D bodies », *in* : *Journal of Computational Physics* 1.*15* (1974), p. 299–319.

[126] Joe F Thompson, Bharat K Soni et Nigel P Weatherill, « Handbook of grid generation », *in* : CRC press, 1998, chap. 22.

[127] JY Tu et L. Fuchs, « Calculation of flows using three-dimensional overlapping grids and multigrid methods », *in* : *International Journal for Numerical Methods in Engineering* 38.*2* (1995), p. 259–282, issn : 1097-0207.

[128] S. Tuann et M. D. Olson, « Numerical studies of the flow around a circular cylinder by a finite element method », *in* : *Computers & Fluids* 6.*4* (1978), p. 219–240.

[129] H.S. Udaykumar et al., « A sharp interface Cartesian grid method for simulating flows with complex moving boundaries », *in* : *Journal of Computational Physics* 174.*1* (2001), p. 345–380.

[130] S. Unverdi et G. Tryggvason, « A front-tracking method for viscous, incompressible, multifluid flows », *in* : *Journal of Computational Physics* 100 (1992), p. 25–42.

[131] Y. l. Utturkar et al., « Sensitivity of synthetic jets to the design of the jet cavity », *in* : *AIAA paper* 124 (2002), p. 2002.

[132] R. Verzicco et al., « Large eddy simulation of a road vehicle with drag-reduction devices », *in* : *AIAA journal* 40.*12* (2002), p. 2447–2455.

[133] R. Verzicco et al., « LES in complex geometries using boundary body forces », *in* : *Center for Turbulence Research Proceedings of the Summer Program. NASA Ames. Stanford University* (1998), p. 171–186.

[134] J. Vos et al., « NSMB Handbook version 4.0 », *in* : (1997).

[135] Z J Wang, « A fully conservative interface algorithm for overlapped grids », *in* : *Journal of Computational Physics* 122 (1995), p. 96–106.

[136] Carlos Weber, « Développement de méthodes implicites pour les équations de Navier-Stokes moyennées et la simulation des grandes échelles : Application à l'aérodynamique externe », Toulouse, INPT, thèse de doct., 1998.

[137] C.H.K. Williamson et R. Govardhan, « Dynamics and forcing of a tethered sphere in a fluid flow », *in* : *Journal of Fluids and Structures* 11.*3* (1997), p. 293–305.

[138] P Wriggers et A Boersma, « A parallel algebraic multigrid solver for problems in solid mechanics discretisized by finite elements1 », *in* : *Computers & structures* 69.*1* (1998), p. 129–137.

[139] JA Wright et W. Shyy, « A pressure-based composite grid method for the Navier-Stokes equations », *in* : *Journal of computational physics* 107 (1993), p. 225–238.

[140] T. Ye et al., « An accurate Cartesian grid method for viscous incompressible flows with complex immersed boundaries », *in* : *Journal of computational physics* 156.*2* (1999), p. 209–240.

[141] Y. Zang et RL Street, « A composite multigrid method for calculating unsteady incompressible flows in geometrically complex domains », *in* : *International Journal for Numerical Methods in Fluids* 20 (1995), p. 341–362.

[142] L. Zhu et C.S. Peskin, « Interaction of two filaments in a flowing soap film », *in* : *Physics of Fluids* 15 (2003), p. 128–36.

[143] Olgierd C Zienkiewicz et Jian Z Zhu, « A simple error estimator and adaptive procedure for practical engineerng analysis », *in* : *International journal for numerical methods in engineering* 24.*2* (1987), p. 337–357.

[144] D Zuzio et JL Estivalezes, « An efficient block parallel AMR method for two phase interfacial flow simulations », *in* : *Computers & Fluids* 44.*1* (2011), p. 339–357.

Anthony Ponce

Raffinement de maillage adapté à la méthode Chimère

Résumé

L'essentiel du travail effectué consiste en l'intégration au sein du solveur NSMB *d'une méthode de raffinement de maillage à partir d'ajout de blocs chimère. Dans un premier temps j'ai du adapter de la structure de données du code existant afin d'ajouter la modularité nécessaire à l'intégration de la méthode. Dans un second temps j'ai implanté la création de blocs de maillages chimères raffinés au cours de l'exécution du calcul. Le raffinement est modulable en fonction de paramètres d'entrées utilisateur. L'ajout de nouveaux blocs de maillages chimère peut être réalisé de manière séquentielle ou parallèle grâce à la parallélisation* openMPI. *Les maillages raffinés peuvent permettre d'affiner une partie de l'écoulement ou s'adapter à une géométrie plus complexe. Ce travail de développement a ensuite été éprouvé et validé sur différentes configurations planes.*

Mots clés : *CFD, Navier-Stokes, OpenMPI, raffinement de maillage, IBM, Chimère*

Summary

The main part of this work consists in the integration into the solver NSMB *of a mesh refinement method based on the addition of chimera blocks. First, i had to adapt the data structure of the existing code in order to add the necessary modularity to integrate the method. In a second step i implemented the creation of refined chimera mesh blocks during the execution of the calculation. The refinement can be adjusted according to user input parameters. The addition of new chimera mesh blocks can be done sequentially or in parallel thanks to the parallelization* openMPI. *Refined mesh sizes can be used to refine part of the flow or to adapt to more complex geometry. This development work was then tested and validated on different flat configurations.*

Résumé EN

Keywords : *CFD, Navier-Stokes, OpenMPI, mesh refinement, IBM, Chimère*